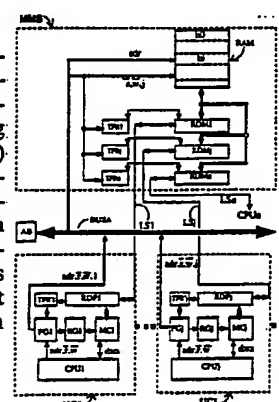




DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITE DE COOPERATION EN MATIÈRE DE BREVETS (PCT)

(51) Classification internationale des brevets⁴ : G06F 12/08, G11C 8/00	A1	(11) Numéro de publication internationale: WO 89/ 06013 (43) Date de publication internationale: 29 juin 1989 (29.06.89)
(21) Numéro de la demande internationale: PCT/FR88/00608 (22) Date de dépôt international: 9 décembre 1988 (09.12.88) (31) Numéro de la demande prioritaire: 87/18103 (32) Date de priorité: 14 décembre 1987 (14.12.87) (33) Pays de priorité: FR (71) Déposant (pour tous les Etats désignés sauf US): CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE (CNRS) [FR/FR]; 15, quai Anatole-France, F-75700 Paris (FR). (72) Inventeurs; et (75) Inventeurs/Déposants (US seulement): LITAIZE, Daniel [FR/FR]; 12, rue de la Chênaie, F-31650 Saint-Orens-de-Gameville (FR). SALINIER, Jean-Claude [FR/FR]; 19, avenue Gleyze-Vieille, F-31520 Ramonville-Saint-Agne (FR). MZOUGH, Abdelaziz [TN/FR]; 80, rue Aristide-Maillo, Bât. B - Apt. 41, F-31100 Toulouse (FR).		ELKHLIFI, Fatima-Zahra [MA/FR]; 9, rue Alain-Lesage, F-31400 Toulouse (FR). LALAM, Mustapha [DZ/FR]; Chez M. & Mme Vautrin, 14, rue des Abeilles, F-31000 Toulouse (FR). SAINRAT, Pascal [FR/FR]; 28 bis, rue des Salenques, Apt. 24, F-31000 Toulouse (FR). (74) Mandataire: BARRE, Philippe; Cabinet Barre-Gatti-Laforgue, 95, rue des Amidonniers, F-31069 Toulouse Cédex (FR). (81) Etats désignés: AT (brevet européen), BE (brevet européen), CH (brevet européen), DE (brevet européen), FR (brevet européen), GB (brevet européen), IT (brevet européen), JP, LU (brevet européen), NL (brevet européen), SE (brevet européen), US. Publiée <i>Avec rapport de recherche internationale.</i>
(54) Title: PROCESS FOR EXCHANGING INFORMATION IN A MULTIPROCESSOR SYSTEM (54) Titre: PROCEDE D'ECHANGE D'INFORMATION DANS UN SYSTEME MULTIPROCESSEUR (57) Abstract <p>A multiprocessor system comprises a core memory (RAM), processing units (CPU₁-CPU_n), each provided with a cache memory (MC_i), a directory (RG_i) and a management processor (PG_i). The core memory (RAM) is connected to an assembly of shift registers (RDM₁-RDM_n) in such a way as to permit, in a cycle of said memory, a parallel transfer, by reading or writing, of data blocks. Each cache memory (MC_i) is connected to a shift register (RDP_i) in such a way as to permit a parallel transfer by reading or writing, of data blocks. An assembly of series connections (LS₁-LS_n) is provided between the assembly of memory shift registers and the assembly of processor shift registers to permit the transfer of data blocks between each pair of associated registers (RDM_i-RDP_i). The addresses of the data blocks can be transmitted between processor (CPU_i) and core memory (RAM) either by said series connections or by a common address bus (BUS A). The architecture according to the invention makes it possible to provide a large number of processing units while obtaining a high output from each processor.</p>  <p>(57) Abrégé</p> <p>L'invention concerne un système multiprocesseur du type comprenant une mémoire centrale (RAM), des processeurs de traitement (CPU₁-CPU_n), chacun doté d'une mémoire-cache (MC_i), d'un répertoire (RG_i) et d'un processeur de gestion (PG_i). La mémoire centrale (RAM) est connectée à un ensemble de registres à décalages (RDM₁-RDM_n) de façon à permettre, en un cycle de cette mémoire, un transfert parallèle en lecture ou écriture des blocs d'informations. Chaque mémoire-cache (MC_i) est reliée à un registre à décalage (RDP_i), de façon à permettre un transfert parallèle en lecture ou écriture des blocs d'informations. Un ensemble de liaisons séries (LS₁-LS_n) est prévu entre l'ensemble des registres à décalage mémoires et l'ensemble des registres à décalage processeurs, pour permettre le transfert de blocs d'informations entre chaque paire de registres associés (RDM_i-RDP_i). Les adresses des blocs d'informations peuvent être transmises entre processeur (CPU_i) et mémoire centrale (RAM), soit par l'entremise de ces liaisons séries, soit par un bus commun d'adresses (BUS A). L'architecture conforme à l'invention permet de prévoir un nombre élevé de processeurs de traitement, tout en bénéficiant d'un rendement élevé pour chaque processeur.</p>		

UNIQUEMENT A TITRE D'INFORMATION

Codes utilisés pour identifier les Etats parties au PCT, sur les pages de couverture des brochures publiant des demandes internationales en vertu du PCT.

AT	Autriche	FR	France	ML	Mali
AU	Australie	GA	Gabon	MR	Mauritanie
BB	Barbade	GB	Royaume-Uni	MW	Malawi
BE	Belgique	HU	Hongrie	NL	Pays-Bas
BG	Bulgarie	IT	Italie	NO	Norvège
BJ	Bénin	JP	Japon	RO	Roumanie
BR	Brésil	KP	République populaire démocratique de Corée	SD	Soudan
CF	République Centrafricaine	KR	République de Corée	SE	Suède
CG	Congo	LI	Liechtenstein	SN	Sénégal
CH	Suisse	LK	Sri Lanka	SU	Union soviétique
CM	Cameroun	LU	Luxembourg	TD	Tchad
DE	Allemagne, République fédérale d'	MC	Monaco	TG	Togo
DK	Danemark	MG	Madagascar	US	Etats-Unis d'Amérique
FI	Finlande				

Procédé d'échange d'information dans un système multiprocesseur.

5

L'invention concerne un système multiprocesseur de type comprenant une mémoire centrale, des processeurs de traitement et des mémoires-caches associées aux processeurs de traitement. Elle s'étend à un procédé d'échange
10 d'informations entre mémoire centrale et processeurs de traitement via la mémoire-cache associée à chacun de ces processeurs. Elle vise également un nouveau composant circuit intégré, susceptible d'équiper le système multiprocesseur.

On sait que, dans les systèmes
15 multiprocesseurs connus les plus usuels, toutes les informations (données, adresses) transitent par un bus commun parallèle de communication entre la mémoire centrale et les divers processeurs de traitement, qui constitue un goulot d'étranglement : son débit est en effet insuffisant pour
20 alimenter tous les processeurs à plein rendement, à partir d'une mémoire centrale commune.

Pour augmenter le débit de transfert d'informations, une première solution consiste à associer à chaque processeur de traitement une mémoire-cache qui, par la
25 localité de l'information, permet de réduire les demandes vers la mémoire centrale. Toutefois, dans le cas où le volume de données partagées entre processeurs est substantiel, le maintien de la cohérence des données entre mémoires engendre un trafic d'informations complémentaire sur le bus de
30 communication qui s'oppose à une réduction significative du débit global sur ce bus et, donc, enlève une grande partie de son intérêt à cette solution.

Une autre solution consiste à réaliser le bus de communication sous la forme d'un réseau maillé désigné par
35 "crossbar", qui permet une communication directe entre chaque processeur de traitement et chaque sous-ensemble de la mémoire centrale (banc-mémoire). Toutefois, cette solution est très lourde et très coûteuse à réaliser du fait du nombre très élevé d'interconnexions, et elle devient totalement irréaliste
40 au-delà d'une dizaine de processeurs de traitement. De plus,

2

en cas de demandes multiples de plusieurs processeurs sur un même banc-mémoire, une telle solution implique des conflits d'accès, source de ralentissement des échanges.

5 Une autre solution plus courante en raison de sa simplicité architecturale consiste à associer une mémoire locale à chaque processeur de traitement pour stocker des données spécifiques à celui-ci, et à mémoriser les données partagées dans la mémoire centrale commune. Toutefois, le gros
10 défaut de cette architecture est sa non-transparence, c'est-à-dire la nécessité pour le programmeur d'organiser le détail des affectations des données dans les diverses mémoires, de sorte que cette solution est d'utilisation très contraignante. De plus, en cas de volume élevé de données
15 partagées, elle peut conduire comme précédemment à une saturation du bus d'accès à la mémoire centrale.

Par ailleurs, une solution dite "architecture aquarius" a été proposée par l'Université de Berkeley et consiste à améliorer la solution crossbar précitée en
20 combinant au réseau crossbar, d'une part, pour les données non partagées, des mémoires-caches qui sont connectées au réseau crossbar, d'autre part, pour les données partagées, des mémoires-caches distinctes qui sont connectées à un bus commun de synchronisation. Cette solution apporte un gain en rapidité
25 d'échanges mais demeure très lourde et très coûteuse à réaliser.

La présente invention se propose de fournir une nouvelle solution, permettant d'augmenter considérablement les débits d'échange d'informations, tout en gardant une
30 architecture transparente pour l'utilisateur, beaucoup plus simple que l'architecture crossbar.

Un objectif de l'invention est ainsi de permettre d'augmenter notablement le nombre de processeurs de traitement du système, tout en bénéficiant d'un rendement
35 élevé pour chaque processeur.

Un autre objectif est de fournir une structure de composant circuit intégré, permettant une réalisation très simple de l'architecture de ce nouveau système multiprocesseur.

40 A cet effet, le système multiprocesseur visé

par l'invention est du type comprenant une mémoire centrale (RAM) organisée en blocs d'informations (bi), des processeurs de traitement ($CPU_1... CPU_j... CPU_n$), une mémoire-cache (MC_j)
5 reliée à chaque processeur de traitement (CPU_j) et organisée en blocs d'informations (bi) de même taille que ceux de la mémoire centrale, un répertoire (RG_j) et son processeur de gestion (PG_j) associés à chaque mémoire-cache (MC_j), des
10 (CPU_j) et mémoire centrale (RAM) ; selon la présente invention, ledit système multiprocesseur est doté :

. d'un ensemble de registres à décalage, dit registres à décalage mémoire ($RDM_1... RDM_j... RDM_n$), chaque registre (RDM_j) de cet ensemble étant connecté à la mémoire
15 centrale (RAM) de façon à permettre, en un cycle de cette mémoire, un transfert parallèle en lecture ou écriture d'un bloc d'informations (bi) entre ledit registre et ladite mémoire centrale,

. de registres à décalage, dits registres à
20 décalage processeur ($RDP_1... RDP_j... RDP_n$), chaque registre à décalage processeur (RDP_j) étant relié à la mémoire-cache (MC_j) d'un processeur (CPU_j) de façon à permettre un transfert parallèle en lecture ou écriture d'un bloc d'informations (bi) entre ledit registre à décalage (RDP_j) et
25 ladite mémoire-cache (MC_j),

. d'un ensemble de liaisons séries ($LS_1... LS_j... LS_n$), chacune reliant un registre à décalage mémoire (RDM_j) et un registre à décalage processeur (RDP_j) et adaptée pour permettre le transfert de blocs d'informations (bi) entre
30 les deux registres considérés (RDM_j, RDP_j).

Ainsi, dans le système multiprocesseur conforme à l'invention, les échanges entre mémoires-caches et processeurs associés s'effectuent comme dans les systèmes classiques équipés de mémoires-caches. Par contre, les
35 échanges entre mémoire centrale et mémoires-caches s'effectuent de façon entièrement originale.

Chaque transfert de bloc d'informations (bi) depuis la mémoire centrale (RAM) vers la mémoire-cache (MC_j) d'un processeur donné (CPU_j) consiste :

40 . à transférer, en un cycle de mémoire

centrale, le bloc (bi) de ladite mémoire centrale (RAM) vers le registre à décalage mémoire (RDM_j) (de la taille d'un bloc) qui est directement connecté à la mémoire centrale et qui
5 correspond au processeur (CPU_j) considéré,

. à transférer sur la liaison série correspondante (LS_j) le contenu de ce registre à décalage mémoire (RDM_j) vers le registre à décalage processeur (RDP_j) (de même capacité) qui est associé à la mémoire-cache (MC_j) du
10 processeur considéré (CPU_j),

. à transférer le contenu dudit registre à décalage processeur (RDP_j) vers ladite mémoire-cache (MC_j).

Dans le sens opposé, chaque transfert de bloc d'informations (bi) depuis la mémoire-cache (MC_j) d'un
15 processeur donné (CPU_j) vers la mémoire centrale (RAM) consiste :

. à transférer le bloc (bi) de ladite mémoire-cache considérée (MC_j) vers le registre à décalage processeur (RDP_j) qui est associé à ladite mémoire-cache
20 (MC_j),

. à transférer sur la liaison série correspondante (LS_j) le contenu du registre à décalage processeur (RDP_j) vers le registre à décalage mémoire (RDM_j), affecté au processeur considéré (parmi l'ensemble de registres
25 à décalage (RDM₁... RDM_j... RDM_n) connectés à la mémoire centrale (RAM)),

. à transférer, en un cycle de mémoire centrale, le contenu du registre à décalage mémoire (RDM_j) vers ladite mémoire centrale (RAM).

30 Dans ces conditions, le transfert de chaque bloc d'informations (bi) s'effectue, non plus à travers un bus parallèle comme c'est le cas dans les systèmes connus, mais par des liaisons séries à haut débit. Ces liaisons séries permettent d'obtenir des temps de transfert de chaque bloc
35 (bi) comparables et même inférieurs aux temps de transfert dans les systèmes connus à bus parallèle. L'exemple comparatif ci-dessous donné avec des valeurs de paramètre courantes pour la technologie actuelle, illustre clairement ce fait qui semble paradoxal.

40 On suppose que chaque bloc d'informations

(bi) est d'une taille égale à 64 octets.

Dans le système de l'invention, le temps de transfert entre la mémoire centrale et une mémoire-cache se décompose en :

- un temps de transfert mémoire centrale (RAM)/registre à décalage mémoire (RDM_j) : 100 nanosecondes (performance d'une mémoire centrale à accès aléatoire de type courant),
- 10 - un temps de transfert série sur la liaison-série correspondante : $64 \times 8 \times 1/500.10^6$, soit 1 024 nanosecondes, en supposant une fréquence de transfert de 500 Mégahertz (non exceptionnelle avec les technologies actuelles qui autorisent des fréquences atteignant
- 15 3 000 Mégahertz),
- un temps de transfert registre à décalage processeur (RDP_j)/mémoire-cache (MC_j) : 50 nanosecondes (mémoire-cache de type très courant).

Le temps de transfert total d'un bloc est donc de l'ordre de 1 200 nanosecondes (en intégrant des délais d'enchaînement de second ordre).

Dans les systèmes connus à mémoires-caches dans lesquels les échanges d'informations s'effectuent directement en parallèle par mots de 4 octets (systèmes les plus courants conduisant à des bus de type habituel à 32 fils

25 de données), le temps de transfert d'un bloc est égal au temps de transfert des 16 mots de 4 octets qui constituent ce bloc, c'est-à-dire : $16 \times 100 = 1\ 600$ nanosecondes.

On voit donc que, avec des hypothèses moyennes dans les deux solutions, ces temps sont comparables. Or, si l'on compare l'architecture du système conforme à l'invention et celle à bus parallèle commun avec mémoires-caches (première solution évoquée précédemment), l'on constate que :

35 . dans la solution classique (bus commun parallèle), la mémoire centrale et le bus commun sont occupés à 100 % pendant le transfert, puisque des informations circulent entre les deux pendant toute la durée du transfert,

 . dans le système conforme à l'invention, la

40 liaison série est occupée à 100 % pendant le transfert, mais

la mémoire centrale est occupée moins de 10 % du temps de transfert (temps de lecture mémoire et de chargement du registre à décalage mémoire (RDM_j)), de sorte que la mémoire centrale peut servir 10 fois plus de processeurs que dans le cas précédent (l'occupation de la liaison série étant sans importance puisqu'elle est privée et affectée au processeur).

Il faut souligner, par ailleurs, que dans le système de l'invention, chaque liaison série qui relie chaque processeur de façon individuelle à la mémoire centrale est une liaison simple (à un ou deux fils de données), de sorte que le réseau série ainsi constitué n'est pas comparable sur le plan de la complexité avec, par exemple, un réseau crossbar dont chaque liaison est une liaison parallèle à multiplicité de 15 fils (32 fils de données dans l'exemple comparatif précité), avec tous les commutateurs nécessaires.

En outre, comme on le verra plus loin sur les courbes comparatives, le système conforme à l'invention possède des performances très améliorées par rapport aux systèmes traditionnels à bus commun et permet en pratique la mise en oeuvre d'un nombre de processeurs de traitement beaucoup plus élevé (de plusieurs dizaines à une centaine de processeurs) ; ces performances sont compatibles à celles d'un système crossbar, mais le système conforme à l'invention est d'une simplicité architecturale beaucoup plus grande.

Dans le système de l'invention, chaque liaison série peut en pratique être réalisée soit au moyen de deux liens-séries unidirectionnels de transfert bit à bit, soit au moyen d'un seul lien bidirectionnel.

Dans le premier cas, chaque registre à décalage mémoire (RDM_j) et chaque registre à décalage processeur (RDP_j) sont dédoublés en deux registres, l'un spécialisé pour le transfert dans un sens, l'autre pour le transfert dans l'autre sens. Les deux liens-séries unidirectionnels sont alors connectés au registre à décalage mémoire (RDM_j) dédoublé et au registre à décalage processeur correspondant (RDP_j) dédoublé, de façon à permettre, l'un un transfert dans un sens, l'autre un transfert dans l'autre sens.

Ce mode de réalisation à deux liens

unidirectionnels présente l'avantage de ne nécessiter aucune gestion de transfert sur lien, mais l'inconvénient de doubler les ressources nécessaires (lien, registres).

5 Dans le second cas, une logique de validation du sens de transfert est associée au lien bidirectionnel de façon à permettre un transfert alterné dans les deux sens sur ledit lien. Cette logique peut être intégrée au processeur de gestion (PG_j) associé à la mémoire-cache (MC_j) à laquelle est
10 relié ledit lien bidirectionnel.

Bien entendu chaque liaison série peut éventuellement être réalisée avec un nombre plus élevé de liens séries.

Dans le système multiprocesseur conforme à
15 l'invention, les moyens de communication d'adresses peuvent revêtir essentiellement deux formes de réalisation : en premier lieu, ils peuvent consister en un bus de communication parallèle d'adresses de blocs (BUSA), commun à tous les processeurs (CPU_j) et reliant ces derniers et la mémoire
20 centrale (RAM) avec de façon classique un arbitre de bus (AB) adapté pour gérer les conflits d'accès audit bus. Il faut noter que ce bus d'adresses n'est utilisé que pour la communication des adresses de blocs : sur le plan de la
25 parallèle d'adresses des systèmes connus, pour lequel ne se pose aucun problème de saturation, puisqu'il peut être libéré aussitôt après transfert de l'adresse de bloc.

Toutefois, un autre mode de réalisation de ces moyens de communication d'adresses peut être envisagé dans
30 le système multiprocesseur de l'invention, consistant à exploiter les liaisons séries de transfert des blocs d'informations (bi) pour transférer les adresses de ces blocs.

Dans ce cas, un registre à décalage complémentaire (RDC_j) est connecté sur chaque liaison série
35 (LS_j) en parallèle avec le registre à décalage mémoire correspondant (RDM_j) : les adresses transmises par ladite liaison série sont ainsi chargées dans chacun de ces registres complémentaires (RDC_j) ; un arbitre de gestion d'accès (ABM) relié auxdits registres (RDC_j) et à la mémoire centrale (RAM)
40 est alors prévu pour prélever les adresses contenues dans

lesdits registres et pour gérer les conflits d'accès à la mémoire centrale (RAM). Un tel arbitre est de conception connue en elle-même, ce type de conflits d'accès étant
5 maintenant résolu depuis de nombreuses années. Dans ce mode de réalisation, on évite la présence d'un bus parallèle de communication d'adresses, mais les ressources en gestion sont alourdies.

Par ailleurs, le système multiprocesseur
10 conforme à l'invention est particulièrement bien adapté pour gérer de façon performante les problèmes de cohérence des données partagées entre processeurs de traitement. En effet, les solutions classiques pour gérer ces données partagées trouvent leur limite dans les systèmes connus du fait du
15 goulot d'étranglement au niveau de la communication des informations, mais deviennent au contraire parfaitement satisfaisantes et performantes dans le système de l'invention où un tel goulot n'existe plus, de sorte que ce système peut être équipé de moyens de gestion des données partagées de
20 conception analogue à ceux des systèmes connus.

Par exemple, une solution traditionnelle de gestion des données partagées consiste à éviter que celles-ci transitent par les mémoires-caches : de façon classique, une logique de partition (LP_j) est associée à chaque processeur
25 de traitement (CPU_j) en vue de différencier les adresses des données partagées et celles des données non partagées de façon à aiguiller les premières directement vers la mémoire centrale (RAM) et les secondes vers la mémoire-cache correspondante (MC_j).

30 Dans une première version de l'architecture conforme à l'invention, le système comprend :

. un bus spécial de communication parallèle de mots (BUSD) reliant les processeurs (CPU_j) et la mémoire centrale (RAM),

35 . une logique de partition (LP_j), associée à chaque processeur (CPU_j) et adaptée pour différencier les adresses des données partagées et celles des données non partagées de façon à transmettre celles-ci sur les moyens de communication d'adresses avec leur identification,

40 . une logique de décodage (DEC) associée à la

mémoire centrale (RAM) et adaptée pour recevoir les adresses avec leur identification et aiguiller les données en sortie mémoire soit vers le registre à décalage mémoire correspondant 5 (RDM_j) pour les données non partagées, soit vers le bus spécial de communication de mots (BUSD) pour les données partagées.

Cette solution présente l'avantage d'être très simple sur le plan architectural ; la présence du bus 10 spécial de communication parallèle (BUSD) conduit à de meilleures performances par rapport à une solution qui consisterait à utiliser les liaisons séries pour transférer non seulement les blocs de données non partagées mais également les mots de données partagées. Il est à noter que 15 cette dernière solution peut, le cas échéant, être envisagée en cas de faible taux de données partagées.

Dans une autre version, le système est doté d'un bus spécial de communication parallèle de mots (BUSD) et d'un bus spécial commun de communication d'adresses de mots 20 (BUSAM), reliant les processeurs (CPU_j) et la mémoire centrale (RAM). La logique de partition (LP_j) aiguille les adresses des données partagées vers le bus spécial commun (BUSAM) en vue du transfert des données par le bus spécial de mots (BUSD), et aiguille les données non partagées vers les 25 moyens de communication d'adresses (que ceux-ci soient constitués par un bus parallèle de communication ou que la communication s'effectue via les liaisons séries).

La présence d'un bus spécial de communication d'adresses de mots permet, dans cette version, de reculer la 30 limite de saturation des moyens de communication d'adresses, en cas de demandes élevées de données partagées.

Une autre version qui sera préférée en pratique dans le cas où les moyens de communication d'adresses sont constitués par un bus parallèle de communication 35 d'adresses (BUSA), consiste à équiper le système d'un processeur de gestion mémoire (PGM) associé à la mémoire (RAM) et d'un processeur espion de bus (PE_j) associé à chaque processeur de traitement (CPU_j) et au répertoire de gestion correspondant (RG_j) ; le processeur de gestion mémoire (PGM) 40 et chaque processeur espion de bus (PE_j), de structures

connues en soi, sont connectés au bus de communication d'adresses (BUSA) en vue respectivement de surveiller et de traiter les adresses de blocs transmises sur ledit bus de façon à permettre une mise à jour de la mémoire centrale (RAM) et de la mémoire-cache associée (MC_j) en cas de détection d'une adresse de bloc présente dans le répertoire associé (RG_j).

Le processeur de gestion mémoire (PGM) et chaque processeur espion (PE_j) associent des bits d'état à chaque bloc d'informations, les tiennent à jour en fonction de la nature (lecture ou écriture) des demandes de bloc qui transitent sur le bus (BUSA) et assurent la cohérence des données partagées en utilisant ces bits d'états qui leur permettent de forcer ou non l'écriture d'un bloc en mémoire centrale au moment des demandes sur le bus (BUSA).

Dans le cas évoqué précédemment où les communications d'adresses se font par les liaisons séries, la gestion des données partagées peut également être assurée de façon centralisée, par un processeur de gestion mémoire (PGM) associé à la mémoire centrale (RAM) et un processeur de maintien de la cohérence des données partagées (PMC_j) associé à chaque processeur de traitement (CPU_j) et au répertoire de gestion correspondant (RG_j), chaque processeur de maintien de cohérence (PMC_j) étant connecté à un bus de synchronisation (SYNCHRO) piloté par le processeur de gestion mémoire (PGM), de façon à permettre une mise à jour de la mémoire centrale (RAM) et de la mémoire-cache associée (MC_j) en cas de détection d'une adresse de bloc, une mise à jour de la mémoire centrale (RAM) et des mémoires-caches (MC_j) à chaque prélèvement d'adresses dans les registres à décalage complémentaires (RDC_j).

Comme précédemment, cette mise à jour est assurée grâce à des bits d'état associés à chaque bloc d'information par le processeur (PGM).

Il est à noter qu'un bus de synchronisation du type ci-dessus défini peut le cas échéant être prévu sur l'architecture précédente où les adresses de blocs transitent sur un bus commun d'adresses BUSA. Dans ce cas, les processeurs espions (PE_j) sont sollicités par le processeur de

gestion mémoire (PGM) via le bus de synchronisation, et ce, uniquement lorsqu'ils sont concernés par le transfert. On évite ainsi des accès inutiles vers les mémoires-caches ; les 5 processeurs espions deviennent alors passifs (puisque sollicités par le processeur PGM) et on les désignera plutôt par l'expression plus appropriée de "processeur de maintien de la cohérence" selon la terminologie ci-dessus utilisée.

Une autre solution consiste à réserver le bus 10 parallèle de communication d'adresses (BUS_A) au transfert des adresses des blocs de données partagées et à utiliser les liaisons séries pour le transfert des blocs de données non partagées.

Par ailleurs, le système multiprocesseur 15 conforme à l'invention se prête à des regroupements de processeurs de traitement sur une même liaison série, de façon à limiter les liaisons séries et les registres à décalage mémoire correspondant (RDM_j) nécessaires.

Le nombre de registres à décalage mémoire 20 (RDM_j) peut correspondre au nombre de liaisons séries (LS_j), auquel cas chaque registre à décalage mémoire (RDM_j) est connecté de façon statique à une liaison série (LS_j) spécifiquement affectée audit registre.

Le nombre de registres à décalage mémoire 25 (RDM_j) peut également être différent de celui des liaisons-séries (LS_j) et notamment inférieur, auquel cas ces registres sont connectés de façon dynamique aux liaisons séries (LS_j) par l'entremise d'un réseau d'interconnexion.

Comme dans les systèmes classiques, la 30 mémoire centrale (RAM) peut être divisée en m bancs mémoires ($RAM_1 \dots RAM_p \dots RAM_m$) agencés en parallèle. Chaque registre à décalage mémoire (RDM_j) est alors constitué de m registres élémentaires ($RDM_{j1} \dots RDM_{jp} \dots RDM_{jm}$) reliés en parallèle à la liaison série correspondante (LS_j). Cependant, un niveau de 35 parallélisme supplémentaire et une meilleure adaptation électrique ou optique de la liaison sont obtenus dans une variante où chaque banc mémoire RAM_p est relié à chaque processeur CPU_j par une liaison série en point à point LS_{jp} .

Par ailleurs, pour présenter des performances 40 de transfert au moins égales à celles des systèmes classiques

à bus parallèle, le système conforme à l'invention est de préférence synchronisé par une horloge de fréquence F au moins égale à 100 mégahertz. Les registres à décalage mémoire (RDM_j) et registres à décalage processeur (RDP_j) peuvent très simplement être d'un type adapté pour présenter une fréquence de décalage au moins égale à F .

Dans le cas de très hautes fréquences (notamment supérieures à 500 mégahertz avec la technologie actuelle), on peut diviser ces registres en sous-registres de fréquence de décalage plus faible, et les multiplexer.

L'invention s'étend à un composant mémoire multiport série, susceptible d'équiper le système multiprocesseur précédemment défini, en vue d'en simplifier la fabrication. Ce composant, qui peut par ailleurs avoir des applications différentes, est constitué par un circuit intégré comprenant une mémoire à accès aléatoire (RAM) de largeur prédéterminée correspondant à un bloc d'informations (bi), un ensemble de registres à décalage ($RDM_1... RDM_j... RDM_n$), chacun de capacité correspondant à la largeur de la mémoire, un bus parallèle interne (BUSI) reliant l'accès de la mémoire et les registres à décalage, une logique de sélection d'un registre à décalage (LSR) adaptée pour valider la liaison sur le bus interne entre la mémoire et un registre à décalage prédéterminé, et un ensemble de broches externes d'entrée/sortie pour l'entrée d'adresses vers la mémoire (RAM), pour l'entrée d'adresses vers la logique de sélection (LSR), pour l'entrée et la validation de commandes de transfert en lecture ou écriture d'un bloc d'informations (bi) entre la mémoire (RAM) et les registres à décalage (RDM_j), pour l'entrée d'un signal d'horloge vers chaque registre à décalage (RDM_j), pour l'entrée bit à bit d'un bloc d'informations (bi) vers chaque registre à décalage (RDM_j) et pour la sortie bit à bit d'un bloc d'informations de chaque registre à décalage (RDM_j).

Ce composant peut être rendu paramétrable par l'adjonction de registres de configuration ($RC_1, RC_2, ...$) permettant notamment un choix de taille des blocs d'informations (bi) et de divers modes de fonctionnement des registres à décalage.

L'invention ayant été exposée dans sa forme générale est illustrée par la description qui suit en référence aux dessins annexés qui en présentent, à titre non limitatif, plusieurs modes de réalisation ; sur ces dessins qui font partie intégrante de la présente description :

- la figure 1 est un schéma synoptique d'un premier mode de réalisation du système multiprocesseur conforme à l'invention,
- 10 - la figure 2 est un diagramme donnant la courbe calculée de performance de ce système (A) et, à titre de comparaison, la courbe (B) correspondante pour une architecture multiprocesseur conventionnelle à bus commun,
- les figures 3, 4 et 5 sont des schémas logiques détaillés d'unités fonctionnelles du système de la figure 1,
- 15 - la figure 6 est un schéma synoptique d'un autre mode de réalisation du système,
- la figure 7 est un schéma synoptique d'un système du type de celui de la figure 1, doté de moyens de gestion des données partagées,
- 20 - la figure 8 est un schéma logique détail d'un sous-ensemble du système de la figure 7,
- la figure 9 est un schéma synoptique d'un système analogue à celui de la figure 7 avec une variante de moyens de gestion de données partagées,
- 25 - la figure 10 est un schéma synoptique d'un système analogue, doté de moyens de gestion de données partagées différents,
- 30 - les figures 11, 12a, 12b, 12c, 12d, 13, 14, 15, 16, 17 sont des schémas logiques détaillés d'unités fonctionnelles du système processeur de la figure 10,
- la figure 18 est un schéma synoptique d'un système du type de celui de la figure 6, doté de moyens de gestion de données partagées,
- 35 - la figure 19 est un schéma synoptique simplifié d'une variante de système, dans laquelle plusieurs unités centrales partagent un même lien-série,
- la figure 20a est un schéma synoptique d'un mode de réalisation préféré, dans lequel la mémoire centrale
- 40

est organisée en plusieurs bancs mémoires,

- la figure 20b est une variante de l'architecture représentée à la figure 20a,

5 - les figures 21a et 21b schématisent une autre structure de mémoire RAM susceptible d'équiper ledit système,

- la figure 22 est un schéma synoptique présentant la structure d'un composant mémoire multiport
10 série, susceptible d'équiper le système.

Le dispositif présenté sous forme de synoptique à la figure 1 est un système multiprocesseur possédant n processeurs de traitement $CPU_1... CPU_j... CPU_n$. On a représenté à cette figure deux processeurs de traitement
15 CPU_1 et CPU_j avec leur logique associée. Chacun de ces processeurs de traitement est d'un type traditionnel par exemple "MOTOROLA 68020" ou "INTEL 80386"... et peut comporter des ressources locales en mémoires et en interfaces périphériques et être équipé d'un dispositif de mémoire
20 virtuelle.

Le dispositif comprend une mémoire centrale à accès aléatoire RAM réalisée de façon classique à partir de circuits intégrés mémoires : en particulier RAM dynamique "INTEL" "NEC" "TOSHIBA"... de 256 Kbits, 1 Mbits, 4 Mbits...
25 selon l'application. Cette mémoire est organisée en blocs d'informations $b_0... b_i... b_t$ de taille t déterminée (habituellement 256 bits à 2 Kbits) et le front d'accès de ladite mémoire correspond à la taille d'un bloc.

La mémoire centrale est reliée en parallèle à
30 n registres à décalage $RDM_1... RDM_j... RDM_n$ dits registres-mémoire, chaque registre-mémoire ayant la taille t d'un bloc d'informations ; chacun de ces registres est réalisé en technologie très rapide ("ASGA"), un bloc pouvant être chargé ou déchargé en un cycle de la mémoire centrale RAM. Le nombre
35 n de registres est égal au nombre de processeurs CPU_j .

Par ailleurs, une mémoire-cache MC_j est associée de façon connue en soi à chaque processeur CPU_j ; chaque mémoire-cache est constituée de façon classique par une mémoire rapide, à accès aléatoire, de faible capacité par
40 rapport à la mémoire centrale RAM. Un répertoire RG_j et un

15

processeur de gestion PG_j sont de façon traditionnelle reliés à la mémoire-cache et au processeur de traitement pour gérer les informations transitant dans la mémoire-cache.

5 En outre, dans le système de l'invention, un registre à décalage RDP_j dit registre-processeur, est relié par son port parallèle, à chaque mémoire-cache MC_j ; chaque registre-processeur RDP_j est de taille correspondant à celle d'un bloc bi et de structure similaire à celle des registres-
10 mémoire RDM_j .

Chaque registre-mémoire RDM_j est relié par son port série, au port série d'un registre-processeur RDP_j par une liaison série LS_j . Des exemples de réalisation de cette liaison série qui peut comporter un lien bidirectionnel
15 ou deux liens unidirectionnels sont illustrés aux figures 4 et 5. Le contrôle du transfert des blocs bi entre registres correspondants RDM_j et RDP_j est assuré par des logiques de transfert TFR_j et TFR'_j qui sont associées de façon symétrique au registre-mémoire RDM_j et au registre-processeur RDP_j ; un
20 exemple de réalisation de ces logiques de transfert (en elles-mêmes classiques) est détaillé à la figure 3.

L'ensemble mémoire centrale RAM, registres à décalages mémoires $RDM_1 \dots RDM_n$ et logiques de transfert associées $TFR_1 \dots TFR_n$ constituent un ensemble fonctionnel
25 dénommé "mémoire multiport série" MMS. L'ensemble processeur de traitement CPU_j , mémoire-cache MC_j , répertoire de gestion du cache RG_j , processeur de gestion du cache PG_j , registre à décalage processeur RDP_j et logique de transfert associée TFR'_j constitue un ensemble "fonctionnel" dénommé "unité
30 centrale" UC_j .

Par ailleurs, le système comprend des moyens de communication d'adresses de blocs des processeurs CPU_j vers la mémoire centrale RAM, constitués en l'exemple par un bus commun de communication parallèle BUSA sur lequel se
35 connectent les processeurs CPU_j (par l'entremise de leur processeur de gestion PG_j) et la mémoire centrale RAM.

L'accès au bus BUSA est réglementé de façon classique par un arbitre de bus AB.

Le fonctionnement général de l'architecture
40 ci-dessus définie est le suivant :

Un processeur CPU_j exécute un programme propre constitué d'instructions, qui se trouvent sous forme de mots en mémoire centrale RAM avec des extraits dans la mémoire-cache associée MC_j. Sur les instructions du programme, le processeur CPU_j est amené soit à lire des mots de données qui elles-mêmes se trouvent dans la mémoire centrale RAM ou dans la mémoire-cache MC_j sous forme d'extraits, soit à écrire des mots de données dans la mémoire centrale RAM et dans la mémoire-cache MC_j.

Chaque opération d'un processeur CPU_j (désignée "demande") nécessite la fourniture de l'adresse adr du mot concerné, la nature r,w de l'opération (lecture, écriture) et l'échange (data) du mot concerné.

Chaque demande de mot active le processeur PG_j qui consulte alors de façon classique le répertoire du cache RG_j lequel indique si le bloc bi contenant le mot concerné est présent dans la mémoire-cache MC_j et, le cas échéant, le cadre de bloc dans la mémoire-cache où se trouve le bloc recherché.

Si le bloc bi contenant le mot concerné est dans la mémoire-cache MC_j, alors en cas de lecture ce mot est lu dans ladite mémoire-cache et envoyé au processeur CPU_j ; en cas d'écriture le mot fourni par le processeur CPU_j est écrit dans la mémoire-cache : la transaction mémoire est terminée.

Si le bloc contenant le mot concerné n'est pas dans la mémoire-cache MC_j, alors une lecture de bloc bi en mémoire centrale RAM est nécessaire. Deux cas peuvent se produire.

30 Premier cas

La mémoire-cache MC_j dispose d'au moins un emplacement de bloc libre, déterminé par le processeur PG_j à l'aide de bits d'états associés à chaque entrée du répertoire RG_j. Dans ce cas, le processeur PG_j requiert, de façon classique, le bus BUSA en soumettant sa demande à l'arbitre de bus AB. Ce dernier accorde, son tour venu, le bus BUSA au processeur PG_j qui accède en lecture à la mémoire centrale RAM, le bloc lu en mémoire étant chargé dans le registre RDM_j, identifié par l'origine j de l'appel. La fin du cycle de lecture se traduit par la libération du bus BUSA et

l'activation du transfert avec la liaison série LS_j permettent de transférer le contenu du registre-mémoire RDM_j dans le registre-processeur RDP_j . La fin du transfert active l'écriture dans la mémoire-cache MC_j du contenu du registre-processeur dans l'emplacement de bloc réservé à cet effet et la transaction peut se terminer comme précédemment.

Deuxième cas

La mémoire-cache MC_j ne dispose pas d'emplacement de libre alors, par un algorithme classique, un emplacement de cache est rendu candidat pour recevoir le bloc demandé. Deux situations peuvent se rencontrer :

- le bloc contenu dans l'emplacement candidat n'a pas été modifié depuis son installation : il est simplement éliminé en libérant le cadre de bloc par une simple écriture d'un bit d'état dans le répertoire (RG_j) et la transaction peut se poursuivre comme précédemment,

- le bloc contenu dans l'emplacement candidat a été modifié et une mise à jour de la mémoire centrale RAM est nécessaire. Pour ce faire, le processeur de gestion PG_j transfère le bloc candidat dans le registre-processeur RDP_j , active le transfert du registre-processeur RDP_j au registre-mémoire RDM_j , puis requiert le bus commun BUSA en soumettant sa demande à l'arbitre AB. Lorsque l'arbitre accorde le bus au processeur de gestion PG_j , ce dernier active une commande d'écriture qui a pour effet de transférer le contenu du registre-mémoire RDM_j à son emplacement en mémoire centrale RAM. La mise à jour mémoire RAM est terminée et la transaction peut se poursuivre comme précédemment.

Ainsi, dans le dispositif de l'invention, les échanges entre les processeurs de traitement CPU_j et leur mémoire-cache MC_j et logiques associées RG_j , PG_j , s'effectuent de façon classique ; par contre, les transferts de blocs entre mémoire centrale RAM et mémoire-caches MC_j passent, non plus par un bus parallèle commun mais par des liaisons-séries LS_j dédiées à chaque processeur de traitement CPU_j , le bus commun BUSA ne servant qu'au transfert des adresses et ayant ainsi un trafic considérablement allégé.

On sait que, pour les architectures classiques à bus commun, une modélisation étudiée par "PATEL"

(analysis of multiprocessors with private cache, JANAK H. PATEL - IEEE Transactions on computers vol. C.31, N° 4 APRIL 1982) a conduit à la formule approchée suivante donnant le rendement U en fonction du nombre de processeurs en présence :

$$U = \frac{1}{1 + m (W + t_f)}$$

10 où

le rendement U est le taux moyen d'utilisation de chaque processeur de traitement,

m est la probabilité pour qu'un processeur de traitement fasse une requête mémoire, non présente dans sa mémoire-cache

15 (cette probabilité $m = \alpha \cdot P_a$ est proportionnelle à la probabilité d'absence P_a de l'information dans la mémoire-cache et à un facteur α fonction de la puissance du processeur de traitement ramené à un pourcentage de requêtes mémoire),

20 W est le temps moyen d'attente du bus commun, qui est fonction du nombre n de processeurs,

t_f est le temps de transfert d'un bloc de la mémoire centrale vers une mémoire-cache.

Les hypothèses à partir desquelles a été
25 établie cette formule montrent que celle-ci est applicable à l'architecture conforme à l'invention, avec un niveau d'approximation comparable au niveau d'approximation de la formule pour des architectures classiques à bus commun.

Il est ainsi possible de comparer les
30 performances des deux types d'architecture en supposant que les composants communs aux deux architectures sont de caractéristiques identiques.

La figure 2 donne les courbes obtenues du rendement U en fonction du nombre n de processeurs pour les
35 paramètres suivants, les paramètres communs aux deux dispositifs étant identiques, et tous de valeur usuelle :

- taille du bloc $b_i = 64$ octets,
- taille du mot pour le transfert parallèle sur bus commun = 4 octets,

- temps d'accès mémoire centrale RAM = 100 nanosecondes,
- temps du cycle bus BUSA = 50 nanosecondes,
- fréquence de transfert série = 500 Mhz,
- 5 - probabilité d'absence $P_a = 0,005$ (mémoire-cache de 16 Koctets),
- facteur de puissance des processeurs : $\alpha = 0,5$.

On constate en comparant les courbes A (architecture de l'invention) et B (architecture classique) que l'architecture conforme à l'invention possède un rendement nettement supérieur à l'architecture classique ; l'architecture de l'invention permet de mettre en place un nombre de processeurs très supérieur aux architectures classiques à bus commun qui en pratique ne peuvent dépasser la dizaine de processeurs. Par exemple, dans le cas classique, un rendement de 0,75 est obtenu dès le dixième processeur, alors qu'il est obtenu pour plus de 80 processeurs dans le cas de l'invention.

La figure 3 présente un mode de réalisation d'une logique de transfert TFR_j ou TFR'_j permettant de transférer un bloc bi d'information d'un registre-mémoire RDM_j vers un registre-processeur RDP_j (le transfert inverse est assuré par des moyens symétriques non représentés à cette figure). Chaque logique TFR_j ou TFR'_j comporte une partie de contrôle de l'émission $TFRE_j$ et $TFRE'_j$ et une partie de contrôle de la réception $TFRR_j$ et $TFRR'_j$ qui sont activées de façon croisée (émission $TFRE_j$ activée en synchronisme avec la réception $TFRR'_j$). Le système comporte un générateur d'horloge H dont la fréquence fixe la vitesse de transmission et fournit le signal d'horloge h à la partie émission $TFRE_j$ et à la partie réception $TFRR'_j$.

Dans la partie émission $TFRE_j$, un registre décompteur DC recevant par son entrée de chargement $load_2$ le signal de lecture \bar{r} du processeur de gestion PG_j permet de laisser passer $t + 1$ impulsions d'horloge h à travers une porte logique ET1 commandée par un signal de passage à zéro "borrow", la sortie de cette porte ET1 étant reliée à l'entrée de décomptage "down" du décompteur DC et à l'entrée de décalage shift1 du registre-mémoire RDM_j .

Dans la partie réception $TFRR'_j$, une bascule

B est reliée par son entrée de donnée D à la sortie série du registre-processeur RDP_j , l'entrée horloge clk de cette bascule étant reliée à l'horloge H pour recevoir le signal h.

5 Un signal d'initialisation "init" fourni par le processeur de gestion PG_j est relié à l'entrée \bar{S} de la bascule B et à l'entrée de chargement $load3$ du registre-processeur RDP_j . La sortie Q de la bascule transmet un signal de commande fin_transfert à la porte logique ET2, permettant de laisser

10 passer le signal d'horloge h vers l'entrée de décalage shift2 du registre-processeur RDP_j . Ce signal de commande est également délivré vers le processeur de gestion PG_j pour indiquer la fin de transfert de bloc.

Le fonctionnement de l'ensemble est le

15 suivant : le processeur de gestion PG_j , après avoir obtenu l'accès à la mémoire centrale RAM via le bus BUSA, effectue sa lecture de bloc bi en fournissant l'adresse du bloc concerné et le signal de lecture \bar{r} . Ce signal déclenche l'activation de la partie émission $TFRE_j$: le front final du signal de lecture

20 r provoque le chargement du bloc bi dans le registre-mémoire RDM_j en activant le signal $load1$ et le chargement de la valeur $t + 1$, correspondante à la taille en bits du bloc bi plus un bit supplémentaire dit de "start", dans le registre décompteur DC par le signal $load2$; ceci a pour effet de remettre à 1 le

25 signal borrow et d'autoriser l'horloge de transfert H à fournir, à travers la porte logique ET1 conditionnée par ce signal borrow, $t + 1$ impulsions d'horloge h : ces impulsions ont pour effet de décaler par l'entrée shift1 $t + 1$ bits du registre-mémoire RDM_j et de faire atteindre la valeur 0 par

30 l'entrée down au décompteur DC : le signal borrow est remis à zéro et verrouille le fonctionnement de la partie émission $TFRE_j$.

Ainsi, la liaison série LS_j , initialement à l'état repos logique 1, transmet le bit 0 dit de start, puis

35 les t bits du bloc bi, et repasse ensuite à l'état repos logique 1, le dernier bit émis étant la valeur 1 forcée sur l'entrée série du registre-mémoire RDM_j .

En préalable à la demande de lecture, le processeur de gestion PG_j a initialisé la partie réception

40 $TFRR_j$ en activant le signal init qui a pour effet de charger

le registre-processeur RDP_j avec t bits à 1 par l'entrée $load3$ et de mettre la sortie Q de la bascule B à l'état logique 1 par l'entrée \bar{S} . Cette sortie Q valide alors la porte logique ET2 qui laisse passer le signal d'horloge h vers l'entrée shift2 du registre-processeur RDP_j . A chaque impulsion d'horloge ce registre-processeur fournit un bit sur sa sortie série qui est mémorisé dans la bascule B . Le premier bit 0 qui se présente a pour effet de mettre à zéro la sortie Q de la bascule B et verrouiller le signal d'horloge h sur la porte ET2. Ce premier bit 0 étant le bit de start qui précède le bloc bi , ce dernier est donc piégé dans le registre-processeur RDP_j lorsque le processeur de gestion PG_j est prévenu du changement d'état de la bascule B par le signal $fin_transfert$: le processeur de gestion PG_j n'a plus qu'à venir lire ce bloc bi sur la sortie parallèle du registre RDP_j .

L'écriture d'un bloc bi vers la mémoire centrale RAM nécessite la présence d'une logique $TFRE'_j$, identique à la logique $TFRE_j$, associée au registre-processeur RDP_j , et d'une logique $TFRR'_j$, identique à la logique $TFRR_j$, associée au registre-mémoire RDM_j . Dans ce cas, le signal $init$ de la logique $TFRR_j$ est relié au signal d'écriture \bar{w} : la libération du registre-mémoire RDM_j réarme automatiquement la logique de réception $TFRR_j$.

Ce mode de réalisation de la logique de contrôle du transfert n'est qu'un exemple possible : le registre émetteur peut être en décalage permanent lui aussi, et le registre récepteur activé pour t impulsions d'horloge sur détection du bit de start en début de transfert.

L'horloge H peut être connectée aux deux registres, ou deux horloges locales indépendantes peuvent être utilisées, la synchronisation étant obtenue de façon classique par un préambule dit de synchronisation.

Le système représenté à la figure 4 comprend un registre à décalage mémoire dédoublé $RDM1_j$ et $RDM2_j$, un registre à décalage processeur dédoublé $RDP1_j$ et $RDP2_j$, deux liens séries unidirectionnels $LS1_j$ et $LS2_j$, l'un reliant le registre-mémoire $RDM1_j$ au registre-processeur $RDP1_j$ de façon à transmettre le contenu du premier vers le second, l'autre

reliant le registre-mémoire RDM2_j au registre-processeur RDP2_j, de façon à transmettre le contenu du second vers le premier, et des logiques associées pour le contrôle du transfert :
 5 TFRE1_j pour RDM1_j, TFRR2_j pour RDM2_j, TFRE2_j pour RDP2_j, TFRR1_j pour RDP1_j.

Pour lire un bloc d'information bi en mémoire centrale RAM, le processeur de gestion PG_j initialise par le signal init la logique TFRR1_j associée au registre-processeur
 10 RDP1_j puis active sa demande de lecture à la mémoire RAM par le signal de lecture \bar{r} . Ce signal active la logique TFRE1_j associée au registre-mémoire RDM1_j : celle-ci assure le transfert sur le lien LS1_j du bloc bi d'information. La fin du transfert est détectée par la logique TFRR1_j associée au
 15 registre-processeur RDP1_j qui prévient le processeur de gestion PG_j de l'arrivée du bloc bi par le signal fin_transfert. Le processeur de gestion PG_j transfère alors le contenu du registre-processeur RDP1_j dans la mémoire-cache MC_j.

20 Pour écrire un bloc mémoire bi, le processeur de gestion PG_j charge le registre-processeur RDP2_j avec le bloc bi concerné extrait de la mémoire-cache MC_j, ce qui active le transfert de ce bloc sur le lien LS2_j. La logique de transfert TFRR2_j associée au registre-mémoire RDM2_j assure la
 25 bonne réception de ce bloc. Le processeur de gestion PG_j est prévenu de la fin du transfert par le changement d'état du signal borrow issu de la logique de transmission TFRE2_j. Le processeur de gestion PG_j effectue alors sa demande d'écriture qui devient effective lors de l'activation du signal
 30 d'écriture \bar{w} ; celui-ci a pour effet de transférer le contenu du registre RDM2_j dans la mémoire centrale RAM et de réinitialiser pour un prochain transfert la logique TFRR2_j.

Ce dispositif autorise un transfert simultané de blocs dans les deux sens et permet de traiter plus
 35 rapidement les défauts de blocs bi dans la mémoire cache MC_j lorsque cette dernière est saturée ; il autorise également la mise en place d'un mécanisme classique d'anticipation de lecture de blocs.

Dans un autre mode de réalisation présenté à
 40 la figure 5, la liaison LS_j comprend un seul lien bi-

directionnel doté à chaque extrémité d'une logique de validation LV_1 et LV_2 constituée par une porte logique à 2 entrées à collecteur ouvert OC_1 et OC_2 , l'une des entrées 5 étant reliée à la sortie série du registre-mémoire RDM_j pour la porte OC_1 et du registre-processeur RDP_j pour la porte OC_2 , l'autre entrée étant reliée à la sortie Q d'une bascule de commande BC_1 et BC_2 ; chacune de celles-ci est reliée par ses entrées \bar{S} et \bar{R} à la logique de transfert TFR pour la bascule 10 BC_1 et TFR' pour la bascule BC_2 .

Lectures et écritures sont effectuées de façon exclusive, à la seule initiative du processeur de gestion PG_j .

Une lecture mémoire active le signal de 15 lecture \bar{F} qui provoque la mise à 1 de la bascule BC_1 par son entrée \bar{S} , la remise à zéro étant commandée, sur l'entrée \bar{R} , par la logique de transfert TFR à la fin du transfert du bloc.

Une écriture mémoire déclenche un mécanisme identique sur la logique de validation LV_2 .

20 D'autres combinaisons registres/liens sont possibles, et dans le cas d'un lien bi-directionnel, on peut notamment utiliser des registres à décalage bi-directionnels recevant un signal de sens de transfert. Cette solution conduit à l'utilisation de registres à décalage plus complexes 25 en logique, donc a priori moins performants en vitesse de transfert.

La vitesse de transfert devant être très élevée, les registres à décalages RDM_j et RDP_j , leurs logiques de commandes associées TFR et TFR', les logiques de 30 validations LV_1 et LV_2 , sont choisis dans une technologie rapide (ECL, ASGA), et synchronisés par une horloge de fréquence F au moins égale à 100 MHz.

Une autre solution à registres multiplexés présentée à la figure 21 permet, comme on le comprendra plus 35 loin, de réduire considérablement la quantité de logique performante, donc coûteuse, nécessaire.

Le système multiprocesseur de la figure 1 était doté, à la fois, d'un bus commun de communication d'adresses de bloc et de liaisons-séries de transfert de 40 données. La figure 6 présente, en variante, un système

multiprocesseur de même principe général, mais dans lequel données et adresses transitent par les liaisons séries, en l'absence de bus commun.

5 Ce système comprend, en plus des registres-mémoires RDM_j , des registres à décalages complémentaires RDC_j , aptes à mémoriser les adresses des blocs demandés et contrôlés par une logique de type TFR_j . En outre, un arbitre de gestion d'accès ABM est relié à la mémoire centrale RAM et
10 aux registres complémentaires RDC_j par leur sortie parallèle. Chaque logique TFR_j est reliée à cet arbitre ABM de structure classique. Le processeur de gestion PG_j de chaque mémoire-cache MC_j est relié à une partie de l'entrée parallèle du registre-processeur RDP_j , en vue d'avoir accès à celui-ci en
15 écriture.

Pour lire un bloc bi en mémoire centrale RAM, le processeur de gestion PG_j place l'adresse du bloc demandé et la nature de la demande (par un bit de préfixe : 1 = lecture, 0 = écriture) dans la partie qui lui est
20 accessible du registre-processeur RDP_j , ce qui a pour effet d'initialiser le transfert de cette information. La logique de transfert TFR_j détecte la fin de transfert sur le registre complémentaire RDC_j et active une demande d'opération vers l'arbitre ABM ; celui-ci est chargé de sérialiser et de
25 traiter les demandes de lecture de bloc en mémoire centrale RAM en allant lire l'adresse du bloc demandé dans le registre complémentaire RDC_j correspondant à la logique de transfert élue par l'arbitre ABM, puis en allant lire le bloc en mémoire centrale RAM qui sera ensuite chargé dans le registre-mémoire
30 RDM_j et transmis comme précédemment.

Pour écrire un bloc en mémoire centrale RAM, le processeur de gestion PG_j enchaîne la transmission de l'adresse puis du bloc à écrire à travers le registre-processeur RDP_j . Le registre complémentaire RDC_j reçoit ainsi
35 tout d'abord l'adresse et la nature de la demande.

La logique de transfert TFR_j analyse cette demande et valide la réception du bloc dans le registre-mémoire RDM_j du fait de la nature de la demande (écriture). La logique de transfert TFR_j est prévenue de la fin de transfert
40 du bloc bi et transmet alors sa requête de service à l'arbitre

ABM. Cette demande est traitée, son tour venu, par ledit arbitre qui active l'écriture du bloc bi en mémoire.

Par ailleurs, le système multiprocesseur 5 représenté à la figure 7 comprend des moyens de gestion des données partagées, permettant de traiter, de façon statique, le problème classique du maintien de la cohérence des données partagées. Ce système comprend les ressources du système de la figure 1 (mêmes désignations) avec les logiques et les 10 ressources supplémentaires suivantes :

Un bus spécial de communication parallèle de mots BUSD relie les processeurs CPU_j et la mémoire centrale RAM. Une logique de partition LP_j est associée à chaque processeur CPU_j ; chaque logique LP_j est constituée de façon 15 classique par un ensemble de couples registres-comparateurs connectés en parallèle sur le bus adresse adr du processeur CPU_j, en vue de réaliser une partition de l'espace mémoire de la mémoire centrale RAM en zone de données non partagées et de données partagées, ladite logique LP_j délivrant à cet effet un 20 signal p (indiquant la nature des données, partagées ou non). Une logique de décodage DEC est associée à la mémoire centrale RAM, elle-même aménagée pour être commandée en écriture par mot ou par bloc par ladite logique DEC.

La logique de décodage DEC est détaillée à la 25 figure 8 et comporte un décodeur DECL, recevant sur son entrée de donnée la partie adresse mot adrm de l'adresse adr, et relié par son entrée de validation à la sortie d'une porte logique ET3, chaque sortie i dudit décodeur étant reliée à un "buffer" de validation de sortie BFS_i. La porte logique ET3 30 reçoit sur ses entrées le signal p et le signal \bar{r} inversé. Un décodeur DECE est relié lui aussi par son entrée de donnée au bus adrm, et par son entrée de validation à la sortie d'une porte logique ET4, ses sorties étant reliées à un ensemble de portes logiques OUI_i en nombre égal au nombre de mots dans un 35 bloc. La porte logique ET4 reçoit sur ses entrées le signal p et le signal \bar{w} inversé. La sortie de la porte ET4 est également reliée à un ensemble de "buffers" de validation d'entrées BFE₁, BFE_i... La mémoire centrale RAM peut être commandée en écriture par mot. Chaque "tranche" mot ainsi 40 définie a son entrée de commande d'écriture w_i. La sortie de

chaque porte logique OUI_i est reliée à l'entrée w_i de chaque "tranche" mot de la mémoire centrale RAM.

La figure 8 présente en outre le détail de l'adressage des registres mémoires RDM_j , qui comprend en premier lieu un décodeur DECEB relié par son entrée de donnée au bus commun BUSA, en vue de recevoir le numéro j du processeur concerné par la requête de l'unité centrale UC_j ; ce décodeur DECEB est relié par son entrée de validation à la sortie d'une porte logique ET5 et par ses sorties 1, 2... j à des "buffers" de validation $BV_1, BV_j...$ La porte logique ET5 reçoit sur ses entrées les signaux p et \bar{w} inversés. De même, un décodeur DECLB est relié par son entrée de donnée au champ j du bus commun BUSA et par son entrée de validation à la sortie d'une porte logique ET6; les sorties 1, 2... j de ce décodeur DECLB sont reliées aux entrées de chargement ld_1, ld_j des registres à décalages mémoire RDM_j . La porte logique ET6 reçoit sur ses entrées les signaux p et \bar{r} inversés.

Le fonctionnement du système est le suivant : à chaque référence mémoire, le processeur CPU_j fournit une adresse sur son bus adresse adr , et la nature de la requête : lecture \bar{r} ou écriture \bar{w} . Il attend une donnée en cas de lecture et fournit une donnée en cas d'écriture. L'adresse adr traverse la logique de partition LP_j , laquelle indique, par le signal p , si l'adresse adr appartient à une zone de données non partagées ($p = 0$) ou de données partagées ($p = 1$). Dans le premier cas, la demande est aiguillée vers le processeur de gestion PG_j et est traitée selon le mode de fonctionnement décrit en référence à la figure 1. Dans le second cas, la requête est directement aiguillée vers le bus commun BUSA; le bus adresse adr comporte des fils supplémentaires d'adresse permettant d'identifier le mot concerné : l'adresse adr est constituée d'une partie adresse bloc $adrb$ et d'une partie adresse mot $adrm$. Ainsi, après accord de l'arbitre de bus AB, la mémoire centrale RAM reçoit soit une demande de transaction bloc ($p = 0$) et dans ce cas, seule la partie bloc $adrb$ de l'adresse adr est significative, soit une demande de transaction mot ($p = 1$) et, dans ce cas, toute l'adresse adr (bloc $adrb$ et mot $adrm$) est significative.

En cas de lecture bloc, $p = 0$ et $r = 0$, la

porte logique ET6 valide le décodeur DECLB qui délivre un signal de chargement LD_j sur le registre à décalage RDM_j , permettant de charger dans ce dernier le bloc lu en mémoire 5 centrale RAM à l'adresse $adrb$ par le signal de lecture \bar{r} .

En cas d'écriture bloc, $p = 0$ et $w = 0$, la porte logique ET5 valide le décodeur DECEB qui délivre un signal de validation sur le "buffer" BV_j , permettant au contenu de ce registre d'être présenté à la mémoire centrale 10 RAM et d'être ainsi écrit à l'adresse $adrb$, la sortie de la porte logique ET5 fournissant le signal d'écriture bloc. Ce dernier est diffusé sur les entrées d'écriture w_1, w_i, \dots aux "tranches" mot de la mémoire centrale RAM à travers les portes logiques OUI_i .

15 En cas de lecture mot, $p = 1$ et $r = 0$, la porte logique ET3 valide le décodeur DFCL qui délivre un signal de validation sur le "buffer" BFS_i , permettant au mot demandé (d'adresse $adrm$ dans le bloc $adrb$) dont la lecture est assurée par le signal \bar{r} , d'être aiguillé vers le bus spécial 20 de communication BUSD. Ce mot est récupéré directement par le processeur CPU_j sur son entrée de donnée data.

En cas d'écriture mot, $p = 1$ et $w = 0$, la porte logique ET4 valide le décodeur DECE qui fournit sur sa sortie i un signal aiguillé à travers la porte logique OUI_i 25 vers l'entrée d'écriture w_i de la "tranche" mot de la mémoire centrale RAM concernée ; ce signal présent à l'entrée w_i permet d'écrire dans cette seule "tranche" mot le mot fourni par le processeur CPU_j sur le bus de donnée BUSD. Le contenu de ce bus est présenté en parallèle sur toutes les "tranches" 30 mot de la mémoire centrale RAM, grâce à une activation des "buffers" BFE_i par le signal issu de la porte logique ET4.

Une caractéristique essentielle de l'architecture de l'invention est de présenter une charge minimale de requêtes sur le bus commun BUSA. Dans 35 l'architecture schématisée à la figure 7, le bus commun BUSA est sollicité par des adresses de bloc et par des adresses de mot. La fréquence des demandes en adresses de mot est fonction du taux de données partagées et peut conduire à une saturation du bus commun BUSA.

40

La figure 9 présente en variante une solution

pour réduire cette charge. Le système visé comprend, en plus des ressources de la figure 7, un bus BUSAM pour les adresses de mot, un arbitre AB' pour arbitrer les conflits d'accès au 5 bus BUSAM, un arbitre ABM chargé d'arbitrer les conflits d'accès en provenance des bus BUSA et BUSAM, et relié à un multiplexeur MUX lui-même relié par ses entrées aux deux bus BUSA et BUSAM.

Le fonctionnement de ce système est le 10 suivant : comme précédemment, la logique de partition LP_j fournit le signal p permettant d'identifier la nature des données manipulées.

Si la demande concerne des données non partagées ($p = 0$), tout défaut d'information entraîne une 15 requête mémoire de type bloc qui transite par le bus commun BUSA.

Si la demande concerne des données partagées ($p = 1$), la requête est aiguillée vers le bus commun BUSAM. Ainsi, la mémoire centrale RAM peut recevoir des demandes 20 simultanées sur les deux bus BUSA et BUSAM, qui doivent donc être arbitrées. L'arbitre ABM alloue, de façon classique, l'accès à la mémoire centrale RAM à l'une des deux requêtes et reconstitue le signal p à partir de l'origine de la demande ($p = 0$ pour BUSA, $p = 1$ pour BUSAM). Le signal p commande 25 alors, d'une part, le multiplexeur MUX qui laisse passer les signaux du bus concerné par la requête, d'autre part, la logique de décodage DEC : on se retrouve dans la situation du système précédent.

On notera que la charge s'est déplacée du bus 30 commun vers la mémoire centrale RAM, puisque le taux de requête au niveau de cette dernière reste le même et que son temps de cycle est du même ordre de grandeur ou même supérieur à celui du cycle bus.

Cette solution n'est donc intéressante que si 35 la mémoire centrale RAM est constituée de bancs mémoire centrale indépendants organisés selon la description donnée plus loin en référence à la figure 20 : plusieurs transactions peuvent dans ce cas, si elles affectent différents bancs mémoires, avoir lieu simultanément.

40

La figure 10 présente un schéma synoptique.

d'un mode de réalisation d'architecture conforme à l'invention, dans lequel le problème des données partagées est traité de façon dynamique. A cet effet, le dispositif conforme à ce mode de réalisation comporte un processeur espion de bus PE_j , couplé à un processeur de gestion du lien parallèle PGP_j . Un processeur de gestion du lien série PGS_j est lié au processeur espion PE_j par une file d'attente $FIFO_j$. Un processeur de gestion de requêtes de l'unité centrale PGU_j est
 10 lié, d'une part, au processeur de traitement CPU_j , d'autre part, aux processeurs de gestion du lien parallèle PGP_j et de gestion du lien série PGS_j . La logique correspondant au processeur de gestion PG_j de chaque mémoire-cache est dans ce mode de réalisation éclatée dans les divers processeurs
 15 présentés ci-dessus. L'accès à la mémoire-cache MC_j et à son répertoire RG_j est règlementé par un processeur de gestion du répertoire et du cache PGR_j .

Enfin, un processeur de gestion PGM de la mémoire centrale RAM est connecté au bus $BUSA$ et à la mémoire
 20 centrale RAM , et à ses registres à décalages RDM_j .

Le fonctionnement de l'ensemble est le suivant :

Chaque transaction sur le bus commun $BUSA$ correspond à une demande de lecture ou d'écriture de bloc bi .
 25 Les processeurs espions de bus PE_j sont activés par chaque demande de lecture de bloc de données. Cette opération réalisée dans le même cycle par tous les processeurs espions va permettre de garantir l'unicité de valeur des données partagées. Le processeur espion PE_j dispose d'un accès au
 30 répertoire RG_j . La fonction d'application utilisée pour la gestion de la mémoire-cache MC_j est dans le mode de réalisation décrit du type application directe. Chaque élément du répertoire est un descripteur de bloc qui contient un champ "tag" (adresse du bloc), des bits classiques d'états du bloc :
 35 un bit de validation v et un bit de modification m et deux bits supplémentaires a pour noter que le bloc est connu de la mémoire-cache mais encore en cours de transfert sur la liaison série, f pour indiquer que le bloc est dans la file d'attente $FIFO$ et éviter ainsi qu'il y soit placé plusieurs fois.

40 Le processeur de gestion mémoire PGM dispose,

d'une part, d'une file d'attente AFIFO d'adresses de blocs bi et d'adresses de processeurs, accessible de façon associative, d'autre part, d'un répertoire d'état des blocs constitué de 5 2 bits par bloc ro et rw indiquant les états possibles de bloc suivant :

. ro = rw = 0 : bloc non encore diffusé,
 . ro = 1 ; rw = 0 : bloc déjà diffusé en
 lecture : une ou plusieurs copies de ce bloc se trouvent dans
 10 les mémoires-caches,

. ro = 0 ; rw = 1 : bloc diffusé en
 écriture : la copie à jour de ce bloc se trouve dans une
 mémoire-cache.

L'évolution des bits d'états de blocs est la
 15 suivante, différente selon la nature de la demande du
 processeur de traitement CPU_j :

- si le processeur CPU_j fait une demande de
 lecture de données non partagées (espace programme ou de
 données explicitement non partagées) : le bloc est marqué déjà
 20 diffusé en lecture (ro = 1 ; rw = 0) côté mémoire centrale
 lors du transfert dudit bloc de la mémoire centrale RAM vers
 le registre RDM_j et marqué non modifié (m = 0) côté mémoire-
 cache dans le même cycle de mise à jour du répertoire RC_j de
 la mémoire-cache (bloc valide). Les espions n'ont pas réagi à
 25 la requête sur le bus commun (la demande ayant été faite avec
 l'indication "données non partagées").

- si le processeur CPU_j fait une demande de
 lecture de données (a priori partagées), le bus commun BUSA
 est occupé pour le temps du passage des informations
 30 d'adresses et de type de requête, le temps de leur traitement
 par le processeur PGM et les espions du bus commun PE_j. En
 mémoire centrale RAM, ce bloc peut être :

1. Non encore diffusé : ro = rw = 0. Il
 est alors transmis à l'unité centrale UC_j et prend l'état non
 35 modifié,

2. Déjà diffusé en lecture : ro = 1 ;
 rw = 0. Il est alors transmis à l'unité centrale UC_j. Son état
 ne change pas,

3. Déjà diffusé en écriture : ro = 0 ;
 40 rw = 1. La copie à jour de ce bloc se trouve dans une mémoire-

cache MC_i . Le processeur espion PE_i associé à cette mémoire-cache a noté la demande de l'unité centrale UC_j lors du passage de l'adresse sur le bus commun et a entrepris son transfert en mémoire centrale RAM dès que possible sur sa liaison série LS_i . En attendant son transfert effectif, le processeur de gestion mémoire PGM met la demande en attente dans la file d'attente associative qui comporte un nombre d'éléments égal au nombre de processeurs.

10 Lors de la demande de lecture sur le bus commun BUSA, tous les processeurs espions PE_i ont réagi en consultant le répertoire RG_i associé à leur mémoire-cache MC_i . Le bus commun BUSA n'est libéré que lorsque tous les processeurs espions PE_i ont eu leur accès au répertoire de gestion RG_i , ce qui garantit le même état du bloc dans tout le système. Le processeur qui possède la copie à jour dans sa mémoire-cache effectue dès que sa liaison série est libre, le transfert de ce bloc dans le registre RDM_i et fait une demande d'écriture de bloc sur le bus commun qui aura pour effet de
15 libérer la demande en attente dans la file associative AFIFO et d'effectuer la mise à jour des bits d'états du bloc.

La mise à jour du bloc n'a donc nécessité qu'une écriture en mémoire centrale RAM sans activation d'espion.

25 - si le processeur CPU_j demande l'écriture d'une donnée dans un bloc présent dans sa mémoire-cache MC_j avec l'état non modifié, une demande d'écriture informative doit être émise sur le bus commun BUSA car il est possible que d'autres mémoires-cache MC_i possèdent ce bloc avec le même état. Ces autres mémoires doivent être informées du changement d'état. A cet effet, tous les processeurs espions PE_i (activés par l'écriture informative diffusée sur le bus commun BUSA) consultent leur répertoire de gestion RG_i et invalident ce bloc, cependant que la mémoire centrale note dans le même
30 temps le changement d'état de ce bloc ainsi que le processeur de gestion parallèle PGP_j , dans le répertoire de gestion RG_j . La libération du bus commun BUSA par tous les processeurs espions et la mémoire centrale RAM permet au processeur CPU_j de réaliser l'écriture dans sa mémoire-cache MC_j , la mise à
40 jour du bit d'état du répertoire de gestion RG_j ayant été

effectuée.

Si une unité centrale est en attente d'accès au bus BUSA pour la même requête sur le même bloc, sa demande se transforme en écriture simple et suit alors le protocole de demande de bloc en écriture.

- si le processeur CPU_j demande l'écriture d'une donnée dans un bloc absent de la mémoire-cache MC_j, ce bloc est lu en mémoire centrale RAM et amené dans la mémoire-cache MC_j pour que l'écriture soit rendue effective.

En mémoire centrale RAM, ce bloc peut être :

1. Non encore diffusé : ro = rw = 0. Le bloc est alors émis sur la liaison série LS_j vers la mémoire-cache MC_j. Il prend les états ro = 0 ; rw = 1 en mémoire centrale et l'état modifié (m = 1) dans la mémoire-cache,

2. Déjà diffusé en lecture : ro = 1 ; rw = 0. Le bloc est émis sur la liaison série LS_j vers la mémoire-cache MC_j. Il prend les états ro = 0 ; rw = 1 en mémoire centrale et l'état modifié (m = 1) dans la mémoire-cache. Lors de la demande sur le bus commun BUSA, les processeurs espions PE_i ont noté la requête et invalidé ce numéro de bloc dans leur mémoire-cache MC_i,

3. Déjà diffusé en écriture : ro = 0 ; rw = 1. La demande est mise dans la file d'attente associative AFIFO et le bus commun BUSA est libéré.

Le processeur espion PE_i de la mémoire-cache MC_i, détenteur de la copie à jour, active dès que possible le transfert du bloc demandé de sa mémoire-cache MC_i vers la mémoire centrale RAM. Ce bloc est ensuite invalidé dans la mémoire-cache MC_i.

L'unité centrale UC_j fait une demande d'écriture de mise à jour de bloc dans les deux cas suivants :

a) la mémoire-cache est saturée et la purge d'un bloc nécessite la mise à jour de ce bloc en mémoire centrale,

b) une unité centrale UC_i est en attente d'un bloc dont la seule copie à jour se trouve dans la mémoire-cache MC_j. Le processeur espion PE_j note la demande et réalise dès que possible la purge de ce bloc.

Côté mémoire centrale RAM, chaque demande

d'écriture de mise à jour entraîne une consultation de la file d'attente associative AFIFO et, en cas de découverte d'une unité centrale UC_i en attente de ce bloc, le chargement de ce 5 bloc dans le registre à décalage RDM_i et la mise à jour des bits d'états correspondant à ce bloc. Ce type de demande d'écriture ne sollicite donc pas les processeurs espions.

Le processeur de gestion du répertoire PCR_j qui est ajouté dans ce mode de réalisation, permet le 10 déroulement de l'algorithme évoqué ci-dessus, en coordonnant les accès au répertoire de la mémoire-cache MC_j qui reçoit des requêtes de trois unités fonctionnelles asynchrones :

1. Le processeur de traitement CPU_j , en vue de lire les instructions du programme en cours d'exécution et 15 de lire ou écrire les données manipulées par ce programme,

2. Le bus commun $BUSA$, en vue de maintenir la cohérence des données dans la mémoire-cache MC_j ,

3. La liaison série LS_j , en vue de charger/décharger un bloc d'information de/vers la mémoire 20 centrale RAM.

Chacune de ces requêtes accède au répertoire de gestion RG_j de la mémoire-cache. La sérialisation de ces accès sur ledit répertoire de gestion permet d'assurer le bon fonctionnement de l'algorithme sus-évoqué de cohérence de 25 l'information dans les mémoires-caches. On obtient ainsi un couplage fort des requêtes au niveau du répertoire de gestion RG_j mais la synchronisation qui doit exister au niveau du traitement de ces requêtes est suffisamment faible pour envisager un fonctionnement asynchrone de la logique de 30 traitement de ces requêtes, ce qui conduit au découpage fonctionnel suivant :

L'interface de chaque processeur CPU_j et de ses auxiliaires (PGS_j , PGU_j) avec le bus commun $BUSA$ est composée de deux parties ayant un fonctionnement mutuellement 35 exclusif : le processeur de gestion du lien parallèle PGP_j est chargé de requérir le bus commun $BUSA$ à la demande du processeur de gestion de requêtes PGU_j ou du processeur de gestion du lien série PGS_j et de piloter le bus commun $BUSA$ en écriture. Le processeur espion de bus PE_j assure la fonction 40 d'espionnage, ce qui revient à piloter le bus commun $BUSA$ en

lecture. Il accède de façon fréquente au répertoire RG_j de la mémoire-cache MC_j .

Le processeur de gestion du lien série PGS_j gère l'interface avec la liaison série LS_j . Il assure le chargement et le déchargement de blocs d'informations bi à la demande du processeur de gestion de requêtes PGU_j et du processeur espion de bus PE_j . Il accède de façon peu fréquente à la mémoire-cache MC_j et au répertoire de gestion RG_j correspondant.

Le processeur de gestion de requêtes PGU_j assure le suivi des requêtes issues du processeur CPU_j . Il accède de façon très fréquente à la mémoire-cache MC_j et au répertoire de gestion RG_j . Cette interface inclut l'éventuelle logique de "MMU" ("Memory Management Unit") habituellement associée au processeur de traitement CPU_j .

Le processeur de gestion PGR_j du répertoire de gestion RG_j est l'arbitre chargé d'allouer l'accès à la mémoire-cache MC_j .

Les figures 11, 12, 13, 14, 15, 16 et 17 représentent à titre d'exemples des modes de réalisation des diverses unités fonctionnelles du dispositif de la figure 10. Les désignations des signaux ou entrées et sorties des unités sont choisies de façon habituelle. Les signaux de même fonctionnalité qui sont engendrés dans chaque unité fonctionnelle à partir d'un signal de base seront désignés par la même référence, par exemple : dnp = données non partagées, dl = demande de lecture, maj = mise à jour, de = demande d'écriture, ei = écriture informative. Le dispositif possède plusieurs processeurs et l'indice j utilisé jusqu'à présent visait un processeur courant et ses auxiliaires ; pour alléger la description, cet indice a été omis sur ces figures et il est bien entendu que la description qui suit vise chacune des unités fonctionnelles qui se rattachent à chaque processeur de traitement. Par ailleurs, les signaux notés x_{YZ} définissent le nom et l'origine du signal dans le cas où $YZ = RG, MC, UC$, et la source et la destination du signal dans les autres cas, avec Y et Z représentant : $U = PGU$, $R = PGR$, $P = PGP$ ou PE , $S = PGS$.

La mémoire-cache MC présentée à la figure 11

a par exemple une capacité de 16 KØ. Elle est organisée en 16 modules de mémoire vive rapide de 1 KØ MC₀, ... MC₁₅, chacun accessible sur un front de 4 octets : le bus adresse de la mémoire-cache MC (noté adr_MC) comporte une partie adresse de bloc adr_bloc et une partie adresse de mot dans le bloc adr_mot. Le bus adresse adr_MC est constitué de 14 fils, permettant d'adresser les 16 KØ de la mémoire-cache MC. La partie adr_bloc comporte 8 fils permettant d'adresser les 256 emplacements de blocs de la mémoire-cache MC, et la partie adr_mot 6 fils permettant d'adresser un mot dans le bloc dont la taille est dans l'exemple de 64 octets.

La partie adresse adr_bloc est reliée à l'entrée adresse de chacun des modules mémoires MC₀... MC₁₅. La partie adresse mot adr_mot est reliée à l'entrée de deux décodeurs DECO et DECI (seuls les 4 bits de forts poids du bus adresse adr_mot sont utilisés : l'adresse est une adresse octet et le cache a une unité d'accès qui est un mot de 4 octets). Le signal de lecture $\overline{r_MC}$ est délivré à chacune des entrées de lecture des modules mémoires MC₀... MC₁₅ et à l'une des entrées d'une porte logique OU1. L'autre entrée de cette porte logique OU1 reçoit le signal \overline{bloc} inversé. Le signal d'écriture $\overline{w_MC}$ est délivré à l'une des deux entrées de portes logiques OU2 et OU3. La porte logique OU2 reçoit sur son autre entrée le signal \overline{bloc} inversé. La porte logique OU3 reçoit sur son autre entrée le signal \overline{bloc} . La sortie de la porte logique OU1 est reliée à l'entrée de validation $\overline{en1}$ du décodeur DECI, et la sortie de rang i de ce décodeur DECI active un "buffer" de validation BVL de rang i. La sortie de la porte logique OU2 est reliée à l'entrée de validation $\overline{en0}$ du décodeur DECO et à des "buffers" de validation BVE. La sortie de la porte logique OU3 est reliée à des portes logiques ET1₀... ET1₁₅, qui reçoivent sur leur autre entrée la sortie de rang correspondant du décodeur DECO. La sortie i de chaque porte logique ET1₀... ET1₁₅ est reliée à l'entrée d'écriture $\overline{w_0}$... $\overline{w_{15}}$ de chaque module mémoire MC₀... MC₁₅. Un bus de donnée relie chaque module mémoire MC₀... MC₁₅ à l'un des buffers de validation BVL et à l'un des buffers de validation BVE. La sortie des buffers BVL et l'entrée des buffers BVE reçoivent en parallèle un bus de donnée datamot_MC (relié au

processeur de gestion des requêtes PGU).

Le fonctionnement de l'exemple de la mémoire-cache ci-dessus décrit est le suivant :

5

Cas 1

La demande provient du processeur de gestion du lien série PGS. Ce cas est signalé par la présence d'un état logique zéro sur le signal bloc.

10

En lecture mémoire-cache, le processeur de gestion du lien série PGS présente sur le bus adresse adr_MC l'adresse de l'emplacement de bloc à lire (dans ce cas, seule la partie adr_bloc du bus adr_MC est utilisée) et active le signal de lecture \overline{T}_{MC} . A l'issue des temps d'accès, le bloc est disponible sur le bus databloc_MC.

15

En écriture mémoire-cache, le processeur de gestion du lien série présente sur le bus adresse adr_MC l'adresse de l'emplacement du bloc à écrire, sur le bus de donnée databloc_MC la donnée à y inscrire, et active la ligne \overline{W}_{MC} . L'état zéro du signal bloc aiguille le signal \overline{W}_{MC} vers les entrées de commande d'écriture des modules de la mémoire-cache MC₀... MC₁₅, via les portes logiques OU3 et ET1₁. L'information présente sur le bus de donnée databloc_MC est inscrite en mémoire-cache à l'issue du temps d'écriture.

25 Cas 2

La demande provient du processeur de gestion de requête PGU du processeur de traitement CPU. Ce cas est signalé par la présence d'un état logique un sur le signal bloc.

30

En lecture mémoire-cache, le processeur de gestion PGU présente sur le bus adr_MC l'adresse du mot demandé, et active le signal de lecture \overline{T}_{MC} . Le bloc correspondant à la partie adr_bloc est lu en mémoire-cache, et le mot demandé est aiguillé, via l'un des buffers de validation BVL, vers le bus de donnée datamot_MC. Le buffer de validation BVL concerné est activé par la sortie du décodeur DECI correspondant à l'adresse mot adr_mot demandée.

35

En écriture mémoire-cache, le processeur de gestion PGU présente sur le bus adr_MC l'adresse du mot à écrire, sur le bus de données datamot_MC la donnée à écrire,

40

et active le signal d'écriture \overline{w}_{MC} . La donnée présente sur le bus datamot_MC est diffusée sur chaque module mémoire-cache, via les buffers BVE validés par le signal d'écriture. Le
5 signal d'écriture \overline{w}_{MC} est ensuite présenté au seul module mémoire concerné. Il est délivré à la sortie du décodeur DECO correspondant à l'adresse adr_mot concernée.

Dans le mode de réalisation ci-dessus décrit, les problèmes d'accès en octet et double octet, et d'accès en
10 double octet et mot à cheval sur deux modules mémoires sont résolus de la même façon que dans les systèmes informatiques traditionnels et ne sont pas décrits ici.

Les figures 12a, 12b, 12c, 12d présentent, à titre d'exemple, les caractéristiques d'un répertoire de
15 gestion du cache RG et d'un processeur de gestion associé PGR. La figure 12a illustre la structure logique de l'adresse adr_RG dans l'hypothèse d'un espace d'adressage sur 32 bits et avec les caractéristiques de la mémoire-cache décrite précédemment. Le champ -tag-, composante de l'adresse bloc,
20 est codé sur 18 bits. Le champ -cadre- est codé sur 8 bits et permet d'adresser les 256 emplacements de bloc de la mémoire-cache MC. Les 6 derniers bits définissent l'adresse mot dans le bloc, en unité octet.

La figure 12b présente la structure du
25 répertoire de gestion du cache RG, qui est une simple mémoire rapide vive de 256 mots de 22 bits. Chaque mot d'adresse i contient le descripteur du bloc inscrit dans l'emplacement i de la mémoire-cache.

La figure 12c schématise la structure du
30 descripteur qui comporte :

- un champ tag de 18 bits, définissant l'adresse du bloc dans l'emplacement ou cadre de bloc courant,
 - le bit de validation v,
 - le bit de modification m,
 - le bit d'attente de fin de transfert a,
 - le bit d'attente de purge f.
- 35

La figure 12d fournit la structure du processeur PGR, qui n'est autre qu'un arbitre classique avec priorité fixe.

40 Cet arbitre comprend un registre LATCH, dont

trois entrées reçoivent respectivement des signaux $\overline{rqst_UR}$, $\overline{rqst_PR}$, $\overline{rqst_SR}$, également délivrés respectivement vers des portes logiques ET2, ET3, ET4. Les sorties correspondantes du registre LATCH sont reliées aux entrées d'un encodeur de priorité PRI, dont les sorties sont reliées aux entrées d'un décodeur DECPRI. Les sorties de rang correspondant à ceux du registre LATCH sont reliées aux signaux $\overline{grnt_UR}$, $\overline{grnt_PR}$, $\overline{grnt_SR}$ ainsi que, de façon inversée, respectivement aux entrées des portes logiques ET2, ET3, ET4. Les sorties des portes logiques ET2, ET3, ET4 sont reliées aux entrées de la porte logique NOU1. La sortie de la porte logique NOU1 est reliée à une bascule B1, qui reçoit sur son entrée D la sortie $\overline{e0}$ de l'encodeur de priorité PRI. L'ensemble du dispositif est synchronisé par une horloge générale qui délivre un signal h à l'une des entrées clk d'une porte logique ET5 et, de façon inversée, sur l'entrée horloge de la bascule B1. La sortie Q de la bascule B1 est reliée à l'autre entrée de la porte logique ET5. La sortie de la porte logique ET5 est reliée à l'entrée load du registre LATCH.

Le fonctionnement de cet arbitre est le suivant : en l'absence de toute requête sur les lignes \overline{rqst} , la bascule B1 mémorise en permanence l'état de la ligne $\overline{e0}$, inactive, et valide ainsi à travers la porte logique ET5 le chargement du registre LATCH.

L'arrivée d'un signal \overline{rqst} provoque le verrouillage de l'horloge et l'activation du signal \overline{grnt} associé au signal \overline{rqst} , jusqu'à désactivation de ce dernier : l'arbitre est figé dans son état pendant toute la durée de la transaction en cours.

Le processeur de gestion des requêtes PGU, représenté à la figure 13, constitue une interface entre le processeur de traitement CPU et :

- d'une part, les divers processeurs avec lesquels il doit échanger des informations : processeur de gestion parallèle PGP, processeur de gestion série PGS, processeur de gestion du répertoire du cache PCR,
- d'autre part, le répertoire de gestion de la mémoire-cache RG et la mémoire-cache MC.

Le processeur de traitement CPU déclenche

l'activité du processeur de gestion des requêtes PGU en activant le signal \overline{as} ("address strobe"). Ce signal valide le bus adresse adr_CPU , les signaux de lecture $\overline{T_CPU}$ et d'écriture $\overline{w_CPU}$ ainsi que les lignes fonction fc_CPU du processeur de traitement CPU. Le processeur de traitement CPU se met alors en attente jusqu'à acquittement de la requête par le signal $\overline{dtack_CPU}$.

Le signal \overline{as} est relié à l'entrée d'un circuit différenciateur D10. La sortie de ce circuit est connectée à l'une des trois entrées d'une porte logique ET12, les deux autres entrées recevant respectivement des signaux $\overline{ack_US}$ et $\overline{ack_UP}$. Ce dernier signal est également délivré à l'entrée \overline{R} d'une bascule B13. L'entrée \overline{S} de la bascule B13 reçoit la sortie d'une porte logique NET10. La sortie de la porte logique ET12 est reliée à l'entrée \overline{S} d'une bascule B11, à l'entrée \overline{S} de la bascule B10 et à l'entrée $\overline{clear10}$ d'un registre à décalages SR10. La sortie \overline{Q} de la bascule B11 fournit le signal $\overline{rqst_UR}$. La bascule B11 reçoit sur son entrée \overline{R} la phase $\theta13$ inversée et la bascule B10 la phase $\theta11$ inversée. La sortie Q de la bascule B10 est reliée à l'entrée série $serial_in10$ du registre SR10. Le registre à décalage SR10 reçoit sur son entrée $clk10$ le signal horloge -h- et sur son entrée de validation $\overline{en10}$ le signal $\overline{grnt_UR}$.

L'activation du signal \overline{as} déclenche le fonctionnement du circuit de différenciation D10. L'impulsion produite par ce circuit traverse la porte logique ET12, met à l'état logique un les bascules B10 et B11 par leur entrée \overline{S} , et effectue également la remise à zéro du registre à décalage SR10 par son entrée $\overline{clear10}$.

La bascule B10 et le registre à décalage SR10 constituent le sous-ensemble logique "distributeur de phases" DP_U. L'activation de ce distributeur de phases est déclenchée par la mise à un de la bascule B10 et la remise à zéro du registre à décalage SR10. Si le registre à décalage est validé par la présence d'un niveau zéro sur son entrée $\overline{en10}$, alors la prochaine impulsion d'horloge h sur l'entrée clk du registre à décalage produit le décalage d'un pas dudit registre.

L'état logique un de la bascule B10 est répercuté sur la sortie $\theta11$ du registre à décalage SR10 par

40

son entrée série serialin10. La sortie 011, appelée phase 011, inversée, remet la bascule B10 à zéro par son entrée \bar{R} . Ainsi, un bit unique est introduit dans le registre à décalages SR10 5 à chaque activation du distributeur de phases DP_U. A chaque impulsion d'horloge h, ce bit va se décaler dans le registre à décalage SR10 et produire les phases consécutives disjointes 011, 012, 013.

La mise à l'état logique un de la bascule B11 10 provoque l'activation du signal $\overline{rqst_UR}$. Ce signal est émis à destination du processeur de gestion du répertoire PGR. Ce dernier, dès que possible, va accorder l'accès au répertoire de gestion RG et à la mémoire-cache MC en activant le signal $\overline{grnt_UR}$, qui va valider l'ensemble des buffers de passage 15 BV10, BV11 et BV12 situés respectivement sur les bus du répertoire de gestion et sur les bus de la mémoire-cache. Ce signal $\overline{grnt_UR}$ valide également le distributeur de phases qui va donc produire séquentiellement les phases 011, 012, 013.

La phase 011 correspond à une temporisation 20 permettant de lire le descripteur du bloc demandé par le processeur de traitement CPU dans le répertoire de gestion RG, adressé par le champ cadre de l'adresse adr_CPU et relié au bus adr_RG à travers des buffers de passage BV10. Le signal $\bar{F_RG}$ est toujours actif à l'entrée d'un buffer BV10, le signal 25 $\bar{W_RG}$ toujours inactif à l'entrée d'un buffer BV10. A l'issue de la temporisation, le descripteur est retourné au processeur PGU via le bus data_RG. La partie tag de ce descripteur et le bit de validation v sont délivrés à l'une des entrées de comparaison d'un comparateur COMP10, l'autre entrée étant 30 reliée à la partie tag de l'adresse adr_CPU. Le bit en regard du bit de validation v est toujours à un. Le comparateur COMP10 est validé en permanence par la présence d'un niveau un sur son entrée en11.

Temps d'accès au répertoire de gestion RG et 35 fréquence d'horloge h sont en rapport tel, qu'à la fin de la phase 011, la sortie egl0 du comparateur COMP10 est positionnée et fournit l'information "le bloc demandé est présent dans la mémoire-cache ou absent de la mémoire-cache".

Si le bloc demandé est présent dans la 40 mémoire-cache MC (egl0 = 1) alors le signal egl0, délivré à

41

l'une des deux entrées de la porte logique ET10, fournit un signal calibré par la phase θ_{12} , présente sur l'autre entrée de la porte logique ET10.

5 Ce signal calibré présent sur la sortie de la porte logique ET10, est relié aux entrées des portes logiques NET10, NET11, NET12.

La porte logique NET10 reçoit sur ses entrées, outre la sortie de la porte logique ET10, le bit 10 inversé d'état m issu du descripteur, et le signal inversé de demande d'écriture $\overline{w_CPU}$.

L'activation de la porte logique NET10 correspond à l'état "demande d'écriture d'un mot dans un bloc présent dans le cache et qui n'a pas encore été modifié 15 ($m = 0$)". La sortie de la porte logique NET10 est reliée à l'entrée \overline{S} d'une bascule B13. L'activation de la porte logique NET10 met la bascule B13 dans l'état logique un, ce qui déclenche une requête d'écriture informative par la ligne $\overline{rqst_UP}$ au processeur de gestion du lien parallèle PGP. 20 L'adresse du bloc concerné est fournie par les lignes adr_bloc_UP , dérivée des lignes adr_CPU .

Le processeur de gestion des requêtes PGU a terminé la première partie de sa tâche : le répertoire de gestion RG et la mémoire-cache sont libérés par désactivation 25 du signal $\overline{rqst_UR}$, conséquence de l'arrivée de la phase θ_{13} inversée sur l'entrée \overline{R} de la bascule B11.

Le mécanisme de l'écriture informative est décrit au paragraphe "processeur de gestion du lien parallèle PGP", et a pour effet de mettre le bloc demandé dans l'état 30 modifié ($m = 1$) ou invalide ($v = 0$). On notera que la libération du répertoire de gestion RG et de la mémoire-cache MC par le processeur de gestion des requêtes PGU est nécessaire afin que le processeur de gestion du lien parallèle PGP puisse y avoir accès. La fin de l'opération "écriture 35 informative" est signalée au processeur de gestion des requêtes PGU par l'activation du signal $\overline{ack_UP}$, qui a pour effet de remettre à zéro la bascule B13 et d'activer, à travers la porte logique ET12, la bascule B11 et le distributeur de phases : le cycle initialement lancé par le 40 signal \overline{as} se reproduit, mais la séquence conséquence de

l'activation de la porte NET10 ne se reproduira pas une seconde fois dans ce cycle de requête.

La porte logique NET11 reçoit sur ses 5 entrées, outre la sortie de la porte logique ET10, le bit d'état m issu du descripteur, et le signal inversé de demande d'écriture $\overline{w_CPU}$.

L'activation de la porte NET11 correspond à l'état "demande d'écriture dans un bloc présent dans le cache 10 et qui a déjà été modifié".

La sortie de la porte NET11 est reliée, à travers un des buffers BV11, au signal d'écriture $\overline{w_MC}$ de la mémoire-cache MC. Ce signal permet d'écrire dans la mémoire-cache, à l'adresse présente sur le bus adr_MC, relié au bus 15 adr_CPU, via les buffers BV11, la donnée présente sur le bus data_MC, relié au bus data_CPU via les buffers bidirectionnels BV12. Le sens d'activation de ces buffers est fourni par le signal $\overline{w_MC}$.

La sortie de la porte NET11 est également 20 reliée à l'une des entrées de la porte logique ET11, qui renvoie ainsi le signal $\overline{dtack_CPU}$ au processeur de traitement CPU. L'opération d'écriture dans le cache se fait en parallèle avec l'activation du signal $\overline{dtack_CPU}$, ce qui est conforme aux spécifications habituelles des processeurs de traitement.

25 L'opération se termine par la libération du répertoire de gestion RG et de la mémoire-cache MC par désactivation du signal $\overline{rqst_UR}$, conséquence de l'arrivée de la phase 013 inversée sur la bascule B11.

La porte logique NET12 reçoit sur ses 30 entrées, outre la sortie de la porte logique ET10, le signal inversé de demande de lecture $\overline{r_CPU}$. L'activation de la porte logique NET12 correspond à l'état "demande de lecture d'un mot dans un bloc présent dans le cache".

La suite des opérations est identique à 35 l'opération précédente, à la seule différence du signal activé ($\overline{r_MC}$ plutôt que $\overline{w_MC}$) associé au sens de transit des données sur les bus data_CPU et data_MC.

Si le bloc demandé est absent de la mémoire-cache (eg10 = 0), alors le signal eg10, inversé, relié à l'une 40 des deux entrées de la porte logique NET13, fournit un signal

calibré par la phase Φ_{12} , présente sur l'autre entrée de la porte logique NET_{13} . La sortie de la porte logique NET_{13} est reliée à l'entrée \overline{S} de la bascule B_{12} . Ce signal calibré force la bascule B_{12} à un, ce qui a pour effet d'émettre une requête de service $\overline{rqst_US}$ vers le processeur de gestion du lien série PGS. Ce processeur reçoit également l'adresse du bloc à requérir sur les lignes adr_bloc_US et la nature de la requête sur les lignes $\overline{w_US}$, $\overline{r_US}$ et fc_US .

10 Le processeur de gestion des requêtes PGU a terminé la première partie de sa tâche : le répertoire de gestion RG et la mémoire-cache MC sont libérés par désactivation de la ligne $\overline{rqst_UR}$, conséquence de l'arrivée de la phase Φ_{13} inversée sur la bascule B_{11} .

15 Le mécanisme de mise à jour du cache est décrit au paragraphe "processeur de gestion de la liaison-série PGS".

On notera que la libération du répertoire de gestion RG et de la mémoire-cache MC est nécessaire pour que 20 le processeur de gestion de la liaison série PGS puisse y avoir accès.

La mise à jour du cache est signalée au processeur de requête PGU par l'activation du signal $\overline{ack_US}$. Ce signal est délivré à l'entrée \overline{R} de la bascule B_{12} et à 25 l'entrée de la porte ET_{12} . Il a ainsi pour effet de remettre à zéro la bascule B_{12} et d'activer à travers la porte logique ET_{12} , la bascule B_{11} et le distributeur de phases : le cycle initialement lancé par le signal \overline{as} se reproduit, mais cette fois avec succès du fait de la présence du bloc dans la 30 mémoire-cache.

Le processeur de gestion série PGS représenté à titre d'exemple à la figure 14 est chargé de gérer la liaison série LS et à ce titre de réaliser les demandes de transfert de blocs entre mémoire centrale RAM et mémoire-cache 35 MC et de réaliser les mises à jour correspondantes dans le répertoire de gestion RG. Il traite en priorité les demandes issues du processeur espion PE en attente dans une file d'attente FIFO. Il traite également les demandes issues du processeur de requêtes PGU.

40 Ce processeur de gestion de la liaison série

PGS comprend une bascule B20 qui reçoit sur son entrée de donnée D le signal empty issu de la file d'attente FIFO et sur son entrée horloge la sortie \bar{Q} d'une bascule B22. Une bascule
5 B21 reçoit sur son entrée de donnée la sortie d'une porte logique OU20. Cette porte logique OU20 valide le signal rqst_US, reçu à l'une de ses deux entrées, l'autre entrée étant reliée au signal empty. L'entrée horloge de la bascule
10 la bascule B22 est rebouclée sur son entrée de donnée D, ce qui la conditionne en diviseur par deux. L'entrée d'horloge de la bascule B22 est connectée à la sortie d'une porte logique ET20, laquelle reçoit sur l'une de ses entrées l'horloge générale de fonctionnement h et sur l'autre entrée un signal
15 de validation. Ce signal de validation provient de la sortie d'une porte logique ET24, qui reçoit respectivement sur ses deux entrées les sorties \bar{Q} et Q des bascules B20 et B21.

La bascule B20 reçoit sur son entrée \bar{R} la phase $\theta 25$, inversée, issue d'un distributeur de phases DP_5.
20 La bascule B21 reçoit sur son entrée \bar{S} la sortie d'une porte logique NOU20. Cette porte logique NOU20 reçoit sur ses deux entrées les sorties respectives de portes logiques ET22 et ET23. La porte logique ET22 reçoit sur ses entrées la phase $\theta 25$ issue du distributeur de phases et le signal maj issu
25 d'une porte logique ET29. La porte logique ET23 reçoit sur ses entrées la phase $\theta 27$ issue du distributeur de phases et le signal maj inversé.

L'ensemble des circuits B20, B21, B22, ET20, ET22, ET23, OU20, ET24, NOU20 constitue un arbitre à priorité
30 fixe ARB_S. Son fonctionnement est le suivant : la bascule B22 fournit sur ses sorties \bar{Q} et Q des signaux alternés de fréquence moitié de celle de l'horloge générale. Ces signaux valident alternativement les bascules B20 et B21. Si une requête de service est présente sur l'une des entrées des
35 bascules B20 ou B21, alors ces signaux alternés mémorisent la demande dans la bascule correspondante (B20 pour le processeur espion PE, B21 pour le processeur de gestion de requête PGU) qui, en retour, verrouille le fonctionnement alterné. On notera que le signal rqst_US en provenance du processeur de
40 gestion de requêtes PGU est conditionné par le signal empty à

travers la porte OU20 : ce signal n'est ainsi pris en compte que si la file d'attente FIFO est vide. La bascule B20 (respectivement B21) n'est remise à zéro que lorsque la 5 transaction à effectuer est terminée.

L'échantillonnage d'une demande sur l'une ou l'autre des bascules B20 et B21 se traduit par un changement d'état de la sortie de la porte logique ET24. La sortie de la porte logique ET24 est également reliée à un circuit 10 différenciateur D20 qui délivre une impulsion lors du changement d'état de la sortie de la porte logique ET24. La sortie du circuit différenciateur est reliée d'une part au distributeur de phases DP_S (entrée \bar{S} d'une bascule B36 et $\text{clr}20$ d'un registre à décalage SR20) du processeur de gestion 15 de la liaison-série et d'autre part à l'une des deux entrées de la porte logique OU22. La sortie de la porte logique est reliée aux entrées \bar{S} de deux bascules B24 et B25. La bascule B24 reçoit sur son entrée \bar{R} la sortie d'une porte logique NOU21 et la bascule B25 le signal grnt_SR . La porte logique 20 NOU21 reçoit sur ses deux entrées les sorties de portes logiques ET36 et ET37. La porte logique E36 reçoit sur ses entrées la phase $\theta 23$ issue du distributeur de phases et le signal maj, et la porte logique E37 la phase $\theta 27$ issue du distributeur de phases et le signal maj inversé.

25 L'impulsion issue du circuit différenciateur D20 initialise ainsi le distributeur de phases et met à l'état logique un les bascules B24 et B25 à travers la porte logique OU22.

Le distributeur de phases DP_S est constitué 30 du registre à décalages SR20 et de la bascule B36. Son fonctionnement est identique à celui décrit dans le paragraphe concernant le processeur de gestion de requêtes PGU.

La sortie \bar{Q} de la bascule B24 est reliée au signal rqst_SR , à destination du processeur de gestion du 35 répertoire PGR. Son activation déclenche une demande de service à ce processeur, qui répond par la ligne grnt_SR , reliée à l'entrée \bar{R} de la bascule B25. La sortie Q de la bascule B25 est reliée à l'une des entrées d'une porte logique OU23. La sortie de la porte logique OU23 est reliée à 40 l'entrée $\text{en}20$ du registre à décalages SR20.

L'ensemble logique B24 et B25 constitue une logique de resynchronisation RESYNC_S entre unités asynchrones. Son fonctionnement est le suivant :

5 Une demande de service rqst_SR vers le processeur de gestion du répertoire PGR se fait par activation des bascules B24 et B25 à travers la porte logique OU22, ce qui autorise deux origines d'activations. La logique propre au processeur de gestion du répertoire PGR assure une
10 réponse grnt_SR dans un temps indéterminé, qui remet à zéro la bascule B25. La bascule B24 maintient sa demande jusqu'à sa remise à zéro par activation de son entrée R. En retour, le processeur de gestion du répertoire PGR désactive sa ligne grnt_SR : la logique de resynchronisation est prête à
15 fonctionner pour une prochaine demande de service. La sortie Q de la bascule B25 sert à bloquer le distributeur de phases par action sur son entrée en20 via la porte logique OU23 : la remise à zéro de la bascule B25 libère le distributeur de phases qui fournira la première phase 021 lors de la prochaine
20 transition active de l'horloge générale h, connectée à l'entrée clk20 du distributeur de phases.

La ligne grnt_SR est également reliée aux buffers de validation BV20, BV25 et BV21, BV22 qui ouvrent l'accès respectivement au répertoire de gestion RG et à la
25 mémoire-cache MC.

Si la bascule B20 est active, alors la transaction en cours est une purge de bloc demandée par le processeur espion PE via la file d'attente FIFO. La sortie Q de la bascule B20 est reliée à des buffers de validation BV24.
30 Ces buffers reçoivent sur leur entrée la sortie d'un registre REG20. La sortie Q de la bascule B20 est reliée à l'une des deux entrées d'une porte logique OU21, qui reçoit sur son autre entrée la sortie du circuit différenciateur D20. La sortie de la porte logique OU21 est reliée à l'entrée load20
35 du registre REG20 et à l'entrée read20 de la file d'attente FIFO.

Ainsi, l'activation de la bascule B20 provoque :

1. L'initialisation du distributeur de
40 phases,

2. Une demande d'accès au répertoire de gestion RG et à la mémoire-cache MC,

3. Le chargement de l'élément en tête de la file d'attente FIFO dans le registre REG20 et l'avance de la file d'attente,

4. La validation du buffer BV24 : le bus adr_X contient l'adresse du bloc à purger. La nature de l'opération (mise à jour) sera retrouvée à partir de la combinaison des bits v et m (signal maj).

Si la bascule B21 est active, alors la transaction en cours provient d'un défaut d'information dans la mémoire-cache MC. La sortie \overline{Q} de la bascule B21 est reliée à des buffers de validation BV23. Ces buffers reçoivent sur leur entrée les informations en provenance du processeur de gestion de requête PGU.

Ainsi l'activation de la bascule B21 provoque :

1. L'initialisation du distributeur de phases,

2. Une demande d'accès au répertoire de gestion RG et à la mémoire-cache MC,

3. La validation des buffers BV23 : le bus adr_X contient l'adresse du bloc qui a provoqué le défaut d'information dans la mémoire-cache MC, et la nature de la requête : lecture ou écriture, données partagées ou non partagées (lignes fc_US).

Le champ "cadre" du bus adr_X est relié aux lignes d'adresses adr_RG du répertoire de gestion RG à travers des buffers de validation BV20. Les sorties \overline{Q} de bascule B26 et B27 sont respectivement reliées aux lignes $\overline{r_RG}$ et $\overline{w_RG}$ du répertoire de gestion à travers les buffers de validation BV20. La bascule B26 reçoit sur son entrée \overline{S} la sortie de la porte logique OU22 et sur son entrée \overline{R} la phase $\theta 22$ inversée issue du distributeur de phases. La bascule B27 reçoit sur son entrée \overline{S} la sortie d'une porte logique NOU22 et sur son entrée \overline{R} la sortie d'une porte logique NOU23. La porte logique NOU22 reçoit sur ses deux entrées les sorties respectives de portes logiques ET25 et ET26, elles-mêmes recevant sur leurs entrées, pour la porte ET25 la phase $\theta 22$ et le signal maj et pour la

porte ET26 la phase $\theta 26$ et le signal maj. La porte logique NOU23 reçoit sur ses deux entrées les sorties respectives de portes logiques ET27 et ET28, elles-mêmes recevant sur leurs 5 entrées, pour la porte ET27 la phase $\theta 23$ et le signal maj inversé et pour la porte ET28 la phase $\theta 27$ et le signal maj inversé.

Le champ tag du bus adr_X est relié au champ tag des lignes data_RG à travers des buffers de validation 10 BV25. Ces derniers reçoivent sur leur entrée de validation la sortie d'une porte logique OU24 qui reçoit sur ses entrées le signal $\overline{\text{grnt_SR}}$ et la sortie \overline{Q} de la bascule B27.

Les entrées D de bascules B28 et B29 sont respectivement reliées aux lignes bit de validation v et bit 15 de modification m du bus data_RG. Les entrées horloges de ces bascules B28 et B29 sont reliées à la phase $\theta 22$ du distributeur de phases. La sortie Q de la bascule B28 et la sortie Q de la bascule B29 sont reliées aux deux entrées d'une porte logique ET29, qui fournit sur sa sortie le signal maj. 20 Une porte logique OU25 reçoit sur ses entrées le signal maj et la sortie \overline{Q} d'une bascule B30. La sortie de la porte logique OU25 est reliée à l'entrée de sélection sel20 d'un multiplexeur MUX20, qui reçoit sur ses deux entrées de données la sortie Q de la bascule B28 (bit v) et la constante 0, la 25 constante zéro étant choisie lorsque l'entrée de sélection sel20 est à l'état logique un. La sortie du multiplexeur MUX20 est reliée à la ligne bit de validation v du bus adr_X. La ligne bit d'attente -a- du bus adr_X est forcée à l'état logique zéro. Le bit de modification -m- est relié à la ligne 30 de lecture $\overline{r_adr_x}$ du bus adr_X.

L'ensemble logique décrit ci-dessus constitue la logique d'accès et de mise à jour du répertoire de gestion RG. Son fonctionnement est le suivant : l'activation du signal $\overline{\text{grnt_SR}}$, autorisant l'accès au répertoire de gestion et à la 35 mémoire-cache, valide les buffers de validation BV20. La lecture du descripteur concerné est commandée du début de l'autorisation d'accès jusqu'à l'arrivée de la phase $\theta 22$, instant de mémorisation des bits -v- et -m- dans les bascules B28 et B29. L'état combiné de ces deux bits produit, à travers 40 la porte logique ET29, le signal maj qui conditionne la suite

des opérations.

1er cas : $\text{maj} = 1$. Ce cas se produit lorsque le bit de validation est égal à 1 et le bit de modification est égal à 1. Ce cas correspond soit à une demande de purge de bloc du processeur espion PE, soit à un défaut d'information constaté par le processeur de gestion de requêtes PGU sur un emplacement de bloc occupé et modifié : dans les deux cas, le bloc concerné doit être écrit en mémoire centrale RAM.

10 A cet effet, le champ -cadre- du bus adr_X est relié aux lignes adr_MC via les buffers de validation BV21. Les sorties \bar{Q} des bascules B30 et B31 sont respectivement reliées aux lignes $\bar{r_MC}$ et $\bar{w_MC}$ de la mémoire-cache MC, à travers les buffers de validation BV21. La ligne
15 bloc est forcée à zéro via un des buffers de validation BV21. Les lignes de données data_MC de la mémoire-cache sont reliées, via les buffers de validation bi-directionnels BV22, aux entrées du registre à décalage RDP et aux sorties des buffers de validation BV26, qui reçoivent sur leurs entrées
20 les lignes de sortie du registre à décalage RDP. Les buffers BV22 sont validés par la ligne grnt_SR , et leur sens de validation commandé par la sortie \bar{Q} de la bascule B31. La bascule B30 reçoit sur ses entrées \bar{S} et \bar{R} respectivement les phases $\theta 21$ et $\theta 23$, inversées, en provenance du distributeur de
25 phases. La bascule B31 reçoit sur ses entrées \bar{S} et \bar{R} respectivement les sorties de portes logiques ET32 et ET33. La porte logique ET32 reçoit sur ses entrées la phase $\theta 26$ et le signal maj inversé, la porte logique ET33 la phase $\theta 27$ et le signal maj inversé. Une porte logique NET20 reçoit sur ses
30 entrées la phase $\theta 23$ et le signal maj . La sortie d'une porte logique ET35 commande les buffers de validation BV26, et reçoit sur ses deux entrées respectivement le signal maj inversé et la sortie Q de la bascule B31. La sortie de la porte logique NET20 est reliée au signal load21 du registre à
35 décalage RDP et à l'entrée \bar{S} d'une bascule B32. L'entrée \bar{R} de cette bascule B32 reçoit le signal fin_transfert_maj issu de la logique TFR associée au registre à décalage RDP. La sortie Q de la bascule B32 est reliée à l'une des entrées de la porte logique OU23.

40

La logique décrite ci-dessus permet de purger

un bloc de la mémoire-cache. Son fonctionnement est le suivant :

En parallèle avec l'accès au répertoire de gestion RG, une lecture de la mémoire-cache MC est activée par la ligne $\overline{r_MC}$, issue de la bascule B30, durant les phases 021 et 022. A l'issue de cette lecture, les données lues, représentant le bloc à décharger, sont présentes à l'entrée du registre à décalage RDM. L'activation du signal maj, dont l'état est connu en début de la phase 022, provoque :

1. L'invalidation du bloc dans le répertoire de gestion : l'entrée sel20 du multiplexeur MUX20 étant à l'état logique un, la valeur zéro est forcée sur le bit de validation, le descripteur étant écrit dans le répertoire de gestion RG avec l'activation du signal $\overline{w_RG}$, commandée par la bascule B27 durant le cycle 022,

2. Le chargement du registre à décalage RDM et l'activation du transfert, lors du passage de la phase 022 à la phase 023,

3. La mise à l'état logique un de la bascule B32, qui va bloquer le distributeur de phases sur la phase 023 jusqu'à la fin du transfert, signalée par le signal fin_transfert_maj issu de la logique TFR,

4. La libération du répertoire de gestion RG et de la mémoire-cache MC par remise à zéro de la bascule B24 sur la phase 023.

Ainsi, l'accès au répertoire est libéré (donc accessible pour le processeur espion PE), le transfert de mise à jour est en cours, et le distributeur de phases bloqué sur 023.

Dès que le transfert est terminé, le bloc est en attente dans le registre à décalage RDM et il faut activer le processeur de gestion du lien parallèle pour que l'écriture en mémoire centrale RAM soit effective.

A cet effet, la bascule B33 est reliée par sa sortie Q à la ligne de requête de service $\overline{rqst_SP}$ à destination du processeur de gestion du lien parallèle PGP. La sortie Q est reliée à l'une des entrées de la porte logique OU23, l'entrée \overline{S} à la phase 024 inversée du distributeur de phases et l'entrée \overline{R} au signal $\overline{ack_SP}$. Le bus $\overline{adr_X}$ est relié

au bus adr_bloc_SP du processeur de gestion du lien parallèle PGP. Une des lignes du bus adr_bloc_SP reçoit le signal maj afin d'indiquer la nature de la requête : mise à jour.

5 Dès libération du distributeur de phases par le signal fin_transfert_maj, la prochaine transition active de l'horloge générale h provoque le passage de la phase 023 à la phase 024. La phase 024 provoque une requête de service au processeur de gestion du lien parallèle PGP (activation de la
10 ligne rqst_SP) et le blocage du distributeur de phases jusqu'à l'acquiescement de la requête par le signal ack_SP. A ce moment, l'écriture en mise à jour aura été effectivement réalisée par le processeur de gestion du lien parallèle PGP. Le distributeur de phases, sur la prochaine transition active
15 de l'horloge h, va passer de la phase 024 à la phase 025.

La mise à jour de la mémoire centrale RAM est terminée : la bascule B20 est mise à zéro par activation de son entrée R par la phase 025 inversée. La bascule B21 est mise à un par activation de son entrée S par la phase 025,
20 conditionnée par le signal maj au moyen de la porte logique ET22 via la porte logique NOU22. En cas de défaut d'information dans la mémoire-cache, la purge d'un bloc ne constitue que la première partie de la requête : la demande rqst_US est toujours présente, mais la libération de la
25 bascule B21 va permettre de prendre en compte d'éventuelles demandes de mises à jour en attente dans la file d'attente FIFO. Dès que la file d'attente FIFO est vide (empty = 0), tout le cycle décrit précédemment se reproduit, avec cette fois le bit de validation à zéro. On se retrouve alors dans le
30 cas suivant.

2ème cas : maj = 0. Ce cas se produit lorsque le bit de validation est égal à zéro (par suite, peut-être, d'une purge de bloc) ou lorsque le bit de validation est égal à un mais le bit de modification est égal à zéro : la copie à jour de ce
35 bloc est déjà en mémoire centrale RAM.

Ainsi, la requête de service rqst_SR va entraîner en retour un accord d'accès au répertoire de gestion RG et à la mémoire-cache MC avec lecture du descripteur, mémorisation des bits m et v, génération du signal maj,
40 réécriture du descripteur avec v = 0 (maj est à l'état logique

zéro, la bascule B31 est à l'état logique zéro et sa sortie Q vaut donc un, ce qui impose un signal logique à l'état un sur l'entrée sel20 du multiplexeur MUX20 et donc force la constante 0) et libération de l'accès au répertoire de gestion RG et de la mémoire-cache MC. Ces opérations sont effectives dès activation de la phase 023.

Le bloc demandé doit maintenant être lu dans la mémoire centrale RAM. A cet effet, le signal maj inversé est reçu à l'une des entrées d'une porte logique NET21, qui reçoit sur son autre entrée la phase 025. La sortie de la porte logique NET21 est reliée aux entrées S des bascules B34 et B35. La bascule B34 reçoit sur son entrée R le signal fin_réception en provenance de la logique de transfert TFR. Sa sortie Q est reliée au circuit différenciateur D21. Ce circuit est relié à la porte logique OU22. L'entrée R de la bascule B35 est reliée au signal grnt_SR et la sortie Q à l'une des entrées de la porte logique OU23.

La lecture d'un bloc en mémoire centrale RAM et son transfert dans la mémoire-cache sont réalisés de la façon suivante : la transition de la phase 023 à la phase 024 déclenche une requête de service au processeur de gestion du lien parallèle par activation de la ligne rqst_SP, issue de la bascule B33. Le type d'opération est cette fois une lecture (r_adr_X = 0) ou une écriture (w_adr_X = 0) et le bus adr_X fournit l'adresse du bloc demandé sur le bus adr_bloc_SP. Le distributeur de phases est bloqué jusqu'à arrivée du signal d'acquiescement ack_SP : la demande de lecture ou d'écriture a été faite à la mémoire centrale RAM par le processeur de gestion du lien parallèle PGP, et le bloc a été en même temps validé et marqué "en attente" par ce même processeur. Le transfert est donc en cours, de la mémoire centrale RAM vers le registre à décalage RDP.

Le distributeur de phases, libéré, fournit ensuite la phase 025. Cette phase, par l'intermédiaire de la porte logique NET21 qui reçoit sur son autre entrée le signal maj inversé, va mettre à l'état logique un les bascules B34 et B35. La bascule B35 bloque le distributeur de phases. La bascule B34 est remise à zéro dès arrivée du bloc demandé dans le registre RDP, signalé par l'activation de la ligne

fin_réception issue de la logique de transfert TFR, ce qui déclenche l'activation du circuit différenciateur D21 et une demande d'accès au répertoire de gestion RG et à la mémoire-cache MC, par activation de la ligne rqst_SR issue de la bascule B24.

L'accord d'accès, signalé par la ligne grnt_SR, libère le distributeur de phases par remise à zéro des bascules B25 et B35 et ouvre l'accès au répertoire de gestion RG et à la mémoire-cache MC. Le distributeur de phases, sur la prochaine transition active de l'horloge générale h, fournit la phase 026.

La bascule B27, reliée au signal \overline{w}_{RG} du répertoire de gestion, est active de 026 à 027, ce qui permet de réaliser la mise à jour du répertoire de gestion avec :

- champ taq du bus data_RG = champ taq du bus adr_X,
- bit de validité $v = 1$ ($maj = 0$ et la sortie Q de la bascule B31 est à zéro : le multiplexeur MUX20 laisse passer v, qui a été forcé à un par le processeur de gestion du lien parallèle PGP, ou déjà remis à zéro par le processeur espion PE),

- bit de modification $m = \text{état de la ligne } \overline{r}_{adr_X} \text{ du bus } adr_X \text{ (demande d'écriture entraîne } m = 1, \text{ demande de lecture } m = 0)$,

- bit d'attente de transfert $a = 0$,
- bit $f = 0$.

La bascule B31 active le signal \overline{w}_{MC} de la mémoire-cache MC de 026 à 027, ce qui permet d'écrire le contenu du registre à décalage RDM (les buffers BV26 sont validés par la sortie \overline{Q} de la bascule B31 et le signal maj inversé) dans le bon emplacement (champ -cadre- du bus adr_X relié au bus adr_MC) de la mémoire-cache, via les buffers de validation BV22, commandés en sens de transfert par la bascule B31.

A l'arrivée de la phase 027, la mise à jour du répertoire de gestion et de la mémoire-cache est terminée. L'arrivée de la phase 027 provoque la remise à zéro de la bascule B24, ce qui libère l'accès au répertoire de gestion RG et à la mémoire-cache MC et l'activation de la sortie de la

porte logique ET23, remettant à un la bascule B21 et activant le signal ack_US à destination du processeur de gestion des requêtes PGU : la requête est terminée.

5 Par ailleurs, le processeur espion PE est chargé du maintien de la cohérence des données dans la mémoire-cache MC ; à titre d'exemple, on a représenté à la figure 15 une architecture de ce processeur. Celui-ci est déclenché à chaque transition active du signal valid du bus
10 commun BUSA.

Si le signal valid est activé par un processeur de gestion du lien parallèle PGP autre que celui associé au processeur espion PE, alors la tâche de ce dernier est la suivante, fonction du type de requête :

15 - requête en lecture de bloc de données non partagées ou requête d'écriture de mise à jour du bloc : néant,

- requête en lecture de bloc de données partagées :

20 . bloc absent : néant,
 . bloc présent et non modifié : néant,
 . bloc présent et modifié : demande de purge du bloc (que ce bloc soit présent ou en cours de transfert de la mémoire centrale RAM vers la mémoire-cache MC,
 25 état signalé par le bit -a- du descripteur),

- requête en écriture de bloc de données partagées :

. bloc absent : néant,
 . bloc présent non modifié : invalider

30 le bloc,

. bloc présent modifié : demande de purge de bloc (même remarque que ci-dessus),

- requête d'écriture informative de bloc :

. bloc absent : néant,
 35 . bloc présent non modifié : invalider

le bloc,

. bloc présent modifié : cas impossible,
 . si une demande d'écriture informative est en attente sur ce même bloc, elle est transformée en
 40 requête en écriture de bloc, car ce bloc a été invalidé ; à

cet effet, la requête en cours est annulée et un acquittement est renvoyé au processeur de gestion des requêtes PGU. Ce dernier consulte le répertoire RG et va trouver le bloc
 5 absent : une requête est émise vers le processeur de gestion PGS. Cette opération est prise en compte par le processeur de gestion parallèle PGP.

Si le signal valid est activé par le processeur de gestion du lien parallèle PGP associé au
 10 processeur espion, alors la tâche de ce dernier est la suivante, fonction du type de requête (dite requête locale) :

- requête en lecture de bloc de données non partagées ou requête d'écriture de mise à jour de bloc : néant,
- 15 - requête en lecture de bloc de données partagées : marquer le bloc valide, en attente de transfert et non modifié ($v = a = 1$ et $m = 0$),
- requête en écriture de bloc de données partagées : marquer le bloc valide, en attente de transfert et
 20 modifié ($v = m = a = 1$),
- requête d'écriture informative de bloc : marquer le bloc "modifié" ($m = 1$).

Pour assurer les fonctions ci-dessus décrites, le processeur espion PE comprend un distributeur de
 25 phases DP_E, constitué d'un registre à décalage SR40 et d'une bascule B40. La sortie d'un circuit différenciateur D40 est reliée à l'entrée \overline{S} de la bascule B40 et à l'entrée clear40 du registre SR40, ainsi qu'à l'entrée \overline{S} d'une bascule B41. La sortie Q de la bascule B40 est reliée à l'entrée serial_in40
 30 du registre SR40. L'entrée clk40 du registre SR40 reçoit le signal h d'horloge générale et l'entrée de validation en40 est reliée à la sortie Q de la bascule B43. La phase A41, inversée, issue du registre SR40 est reliée à l'entrée \overline{R} de la bascule B40.

35 Le fonctionnement du distributeur de phases est conforme à la description faite dans le processeur de gestion des requêtes PGU.

Le signal valid du bus BUSA est relié à l'entrée du circuit différenciateur D40 et à l'entrée de
 40 validation en41 d'un décodeur DEC41. La bascule B41 reçoit sur

son entrée \overline{R} la sortie de la porte logique NOU40. La sortie \overline{Q} de la bascule B41, à collecteur ouvert, est reliée au signal done du bus BUSA.

5 Les signaux valid et done assurent la synchronisation du processeur espion PE avec les autres processeurs espions du système multiprocesseur : la transition négative du signal valid déclenche le circuit différenciateur D40 qui produit une impulsion permettant d'activer le distributeur de phases et de mettre le signal done à l'état 10 logique zéro par l'intermédiaire de la bascule B41. La fin du travail de l'espion est signalée par un changement d'état sur la sortie de la porte logique NOU40, qui produit la mise à l'état logique un du signal done par l'intermédiaire de la 15 bascule B41.

Le travail de l'espion est fonction de la nature de la demande et, à cet effet, le champ type du bus BUSA est connecté à l'entrée du décodeur DEC41. Les sorties dnp et dmaj du décodeur DEC41 sont reliées aux entrées d'une 20 porte logique OU40. Les sorties dl, de, dei du décodeur DEC41 sont reliées aux entrées d'une porte logique OU41, les sorties de et dei étant également reliées aux entrées d'une porte logique OU42. La sortie de la porte logique OU40 est reliée à l'une des entrées de la porte logique NOU40 par 25 l'intermédiaire d'une porte logique ET40, qui reçoit sur son autre entrée le signal 041. La sortie de la porte logique OU41 est reliée respectivement à l'une des entrées de portes logiques ET42, ET43, qui reçoivent sur leur autre entrée respectivement le signal de phase 041 et 044. Les sorties des 30 portes logiques ET42 et ET43 sont reliées respectivement aux entrées \overline{S} et \overline{R} d'une bascule B44. La sortie de la porte logique ET42 est également reliée aux entrées \overline{S} de bascules B42 et B43. Une porte logique NOU41 reçoit sur ses entrées le signal de phase 045 et la sortie d'une porte logique ET45, qui 35 reçoit sur ses deux entrées la phase 044 et la sortie inversée d'une porte logique OU43. La sortie de la porte logique NOU41 est reliée à l'entrée \overline{R} de la bascule B42. La sortie \overline{Q} de la bascule B42 est reliée au signal $\overline{rqst_PR}$ et le signal $\overline{qrnt_PR}$ est délivré à l'entrée \overline{R} de la bascule B43, aux entrées de 40 validation de buffers de passage BV40, et à l'une des entrées

de la porte logique OU44. La porte logique OU44 reçoit sur son autre entrée la sortie \bar{Q} d'une bascule B45, et sa sortie valide les buffers de passage BV41. La sortie de la porte logique ET45 est également reliée à l'une des entrées de la porte logique NOU40 qui reçoit sur une autre entrée le signal de phase $\theta 45$. La sortie de la porte logique OU42 est reliée aux entrées de portes logiques ET46 et ET47, qui reçoivent également sur leurs entrées la sortie d'une porte logique OU43 et respectivement les phases $\theta 44$ et $\theta 45$. La sortie \bar{Q} de la bascule B44 délivre le signal \bar{r}_{RG} via un buffer BV40, la sortie \bar{Q} de la bascule B45 délivre le signal \bar{w}_{RG} via un buffer BV40. Le champ cadre du bus commun BUSA est relié au bus adr_{RG} via un buffer BV40. Le bus $data_{RG}$ est relié aux sorties des buffers de validation BV41 et à l'entrée du registre REG40, qui reçoit sur son entrée $load40$ le signal de phase $\theta 43$. La sortie du registre REG40 est reliée, pour la partie tag, aux entrées des buffers BV41 et à l'une des entrées d'un comparateur COMP40. Le comparateur COMP40 reçoit sur son autre entrée la partie tag du bus BUSA. Le bit de validation v , issu du registre REG40, connecté au comparateur COMP40 a en regard la valeur constante 1. Les bits v , a , m , f en sortie du registre REG40 sont reliés respectivement à l'une des entrées des multiplexeurs MUX40, MUX41, MUX42, MUX43. Les sorties de ces multiplexeurs fournissent l'état de ces mêmes bits à l'entrée des buffers BV41. Le multiplexeur MUX40 reçoit sur ses autres entrées les constantes zéro et un, les entrées $sel40$ sont reliées à la sortie d'une porte logique ET48 et à un signal $d1le$. Le multiplexeur MUX41 reçoit sur son autre entrée la constante un, sélectionnée lorsque son entrée $sel41$, recevant un signal $d1le$ issu du processeur de gestion PGP, est à l'état logique un. Le multiplexeur MUX42 reçoit sur ses autres entrées la constante un et le signal $\bar{r}_{adrbloc_SR}$, ses entrées $sel42$ reçoivent les signaux $d1ei$ et $d1le$ issus du processeur de gestion PGP. Le multiplexeur MUX43 reçoit sur son autre entrée la constante un, sélectionnée lorsque son entrée $sel43$, reliée à la sortie d'une porte logique ET49, est à l'état logique un. La porte logique ET49 reçoit sur ses entrées la sortie $eq40$ du comparateur COMP40, le signal f inversé et le signal m . La porte logique ET48 reçoit sur ses

entrées la sortie eg40, le signal m inversé, le signal dlei et la sortie de la porte logique OU42. La porte logique OU43 reçoit sur l'une de ses entrées le signal eg40, et sur son autre entrée le signal dle. La sortie de la porte logique ET49 est également reliée à l'entrée load41 de la file d'attente FIFO, déjà décrite dans le processeur de gestion du lien série PGS. Les champs cadre et tag du bus BUSA sont reliés à l'entrée de la file d'attente FIFO. Les signaux dle, dle et r_adrbloc_SP proviennent du processeur de gestion du lien parallèle PGP, qui reçoit par ailleurs le signal dei du décodeur DEC41.

Le fonctionnement de l'ensemble est le suivant : l'activation du signal valid initialise le distributeur de phases DP_E et valide le décodeur DEC41 qui produit l'activation d'une sortie fonction de l'information type codant la nature de la demande en cours sur le bus commun BUSA. La sortie active peut être :

- dnp : requête en lecture de données non partagées. Le signal done est désactivé à l'apparition de la phase 041 ;

- dmaj : requête d'écriture de mise à jour de bloc. Le signal done est désactivé à la phase 041 ;

- dl : requête de lecture de bloc ;
- de : requête d'écriture de bloc ;
- dei : requête d'écriture informative.

Ces trois cas nécessitent un accès en lecture au répertoire RG, et les deux derniers une réécriture éventuelle du répertoire. A cet effet, une requête d'accès est émise vers le processeur de gestion du répertoire PGR par la bascule B42 (signal rqst_PR), la bascule B43 inhibant le distributeur de phases jusqu'à accord d'accès, signifié par le signal grnt_PR. La lecture s'effectue donc de 041 à 044 (bascule B44) et l'éventuelle écriture de 044 à 045 (bascule B45) avec mémorisation du descripteur dans le registre REG40 lors de la phase 043. Si le bloc est absent de la mémoire-cache MC (eg40 = 0), alors le signal done est désactivé sur la phase 044. Si le bloc est présent dans le cache (eg40 = 1), alors :

- si m = 1, une demande de purge est activée

(activation porte logique ET49) à condition que ce bloc ne soit pas déjà dans la file ($f = 0$) ; le seul bit modifié est f , mis à un, lors de la réécriture du descripteur,

5 - si $m = 0$, le bloc est invalidé par le multiplexeur MUX40 (activation de la porte logique ET48),

- si la demande est locale (d_{lle} ou d_{lei} actif), alors :

1) en cas de lecture ou d'écriture, les 10 bits v et a sont mis à 1 et m est mis à 0 (lecture) ou 1 (écriture) (état du signal $\overline{r_adrbloc_SP}$),

2) en cas d'écriture informative, le bit m est forcé à 1.

Dans ces derniers cas qui demandent une 15 réécriture dans le répertoire de gestion RG, le signal $done$ est désactivé sur la phase $\theta 45$.

Le processeur de gestion du lien parallèle PGP, dont un exemple est représenté à la figure 16, est chargé de requérir le bus commun BUSA et de réaliser la transaction 20 demandée, soit par le processeur de gestion des requêtes PGU, soit par le processeur de gestion de la liaison série PGS.

Une demande issue du processeur de gestion des requêtes PGU ne peut être qu'une requête d'écriture informative. Une demande issue du processeur de gestion de la 25 liaison série PGS est, soit une demande de lecture ou d'écriture de bloc, soit une demande de mise à jour de bloc.

Le processeur de gestion du lien parallèle PGP comprend une bascule B60, reliée par son entrée de donnée D au signal $\overline{rqst_UP}$. Une bascule B61 est reliée par son entrée 30 de donnée D au signal $\overline{rqst_SP}$. Les sorties Q et \overline{Q} d'une bascule B62 sont reliées respectivement aux entrées horloges des bascules B60 et B61. La sortie \overline{Q} de la bascule B62 est rebouclée sur son entrée de donnée. Les sorties \overline{Q} des bascules B60 et B61 sont reliées aux entrées d'une porte logique OU60. 35 La sortie de la porte logique OU60 est reliée, d'une part, à un circuit différenciateur D60, d'autre part et de façon inversée, à une entrée d'une porte logique ET60, qui reçoit sur son autre entrée le signal d'horloge générale h . La sortie de la porte logique ET60 est reliée à l'entrée horloge de la 40 bascule B62. La sortie du circuit différenciateur D60 est

reliée à l'entrée \overline{S} d'une bascule B63. La sortie \overline{Q} de la
bascule B63 délivre un signal \overline{rqsti} à destination de l'arbitre
AB, et son entrée \overline{R} est reliée à la sortie d'une porte logique
5 ET62. Le signal \overline{grnti} issu de l'arbitre AB est relié à des
portes logiques OU62 et NOU60. La porte logique OU62 reçoit
sur son autre entrée le signal \overline{valid} inversé, et sa sortie est
reliée à l'entrée d'un circuit différenciateur D61. La sortie
de ce circuit D61 est reliée à l'une des entrées de la porte
10 logique ET62 et à l'entrée \overline{S} d'une bascule B64. La sortie Q de
cette bascule B64 est reliée à l'entrée de validation de
buffers de passage BV60 et de façon inversée, à travers un
inverseur à collecteur ouvert I60, au signal \overline{valid} . Le signal
 \overline{done} est relié à l'entrée d'un circuit différenciateur D62. La
15 sortie de ce circuit D62 est reliée à l'entrée \overline{R} de la bascule
B64 et à l'une des entrées de la porte logique NOU60. La
sortie de cette porte logique NOU60 est reliée à l'une des
entrées de portes logiques NET60 et NET61, qui reçoivent sur
leur autre entrée, respectivement les sorties \overline{Q} des bascules
20 B60 et B61. Les sorties \overline{Q} des bascules B60 et B61 sont
également reliées respectivement aux entrées de validation des
buffers de passage BV61 et BV62. La sortie de la porte logique
NET60 est reliée, d'une part, à l'entrée \overline{S} de la bascule B60,
d'autre part, à l'une des entrées d'une porte logique ET63,
25 qui reçoit sur son autre entrée la sortie d'un circuit
différenciateur D63. La sortie de la porte logique ET63
délivre le signal $\overline{acq_UP}$ vers le processeur de gestion PGU. La
sortie de la porte logique NET61 est reliée à l'entrée \overline{S} de la
bascule B61 et fournit le signal $\overline{ack_SP}$. Le bus $\overline{adr_bloc_UP}$
30 est relié à l'entrée des buffers de validation BV61 et à l'une
des entrées d'un comparateur COMP60.

Le bus $\overline{adr_bloc_SP}$ est relié à l'entrée des
buffers de validation BV62. Les sorties des buffers BV61 et
BV62 sont reliées ensemble et à l'entrée des buffers de
35 validation BV60. La sortie des buffers BV60 est reliée au bus
commun BUSA. Des portes logiques OU63 et OU64 reçoivent sur
leurs entrées respectives les sorties \overline{Q} des bascules B60 et
B61, le signal logique \overline{grnti} et le signal \overline{maj} pour OU64. La
sortie de la porte logique OU63 délivre le signal \overline{dlei} , et la
40 sortie de la porte logique OU64 le signal \overline{dlii} . L'autre entrée

du comparateur COMP60 reçoit les champs taq et cadre du bus commun BUSA. L'entrée en60 du comparateur COMP60 est reliée à la sortie de la porte logique ET61, qui reçoit sur ses entrées
5 le signal grnti inversé, le signal dei et le signal rqst_UP inversé. La sortie eg60 du comparateur COMP60 est reliée à l'entrée du circuit différenciateur D63. La sortie de ce circuit est également reliée aux entrées R des bascules B60 et B61 et à l'autre entrée de la porte logique ET62.

10 Le fonctionnement de l'ensemble est le suivant :

Les bascules B60, B61 et B62 associées aux portes logiques ET60 et OU60 constituent un arbitre local. Cet arbitre examine alternativement les requêtes rqst_UP et
15 rqst_SP, et les répercute vers l'arbitre AB du bus commun BUSA par le signal rqsti. L'accord d'accès est fourni par la validation du signal grnti et le cycle bus a lieu dès désactivation du signal valid qui libère l'arbitre AB. L'activation du signal done libère l'arbitre local : la
20 transaction est terminée.

Si la requête provient du processeur de gestion de la liaison série PGS, alors les signaux dlei et dlle indiquent au processeur espion PE associé la nature de la mise à jour des bits d'états du bloc à effectuer dans le
25 répertoire RG.

Si la requête provient du processeur de gestion des requêtes PGU, alors en cas de détection d'écriture informative sur le même bloc (signal dei en provenance du processeur espion PE), une libération immédiate a lieu : le
30 processeur de gestion des requêtes PGU, après consultation du répertoire (le bloc a été invalidé) aiquillera sa demande vers le processeur de gestion de la liaison série PGS.

Le processeur de gestion mémoire PGM dont un exemple est représenté à la figure 17, est chargé d'assurer la
35 lecture ou l'écriture de bloc en mémoire centrale RAM et de participer au maintien de la cohérence de l'information dans les diverses mémoires-caches MC du système multiprocesseur.

A cet effet, il comprend un circuit différenciateur D80 recevant sur son entrée un signal valid et
40 relié par sa sortie aux entrées S de bascules B80 et B81 ainsi

qu'à l'entrée clr_80 d'un registre à décalages SR80. Sur la sortie Q de la bascule B80, à collecteur ouvert, est délivré le signal done. La sortie Q de la bascule B81 est reliée à l'entrée serial_in80 du registre SR80 ; cette bascule B81 est reliée par son entrée R à la sortie Q81, inversée, du registre SR80. Le registre SR80 reçoit sur son entrée clk80 le signal d'horloge générale h et son entrée de validation en80 est toujours active. La bascule B81 et le registre à décalage SR80 constituent un distributeur de phases DP_M. Le signal valid est également délivré sur l'entrée de validation en81 d'un décodeur DEC80. Ce décodeur est relié par son entrée de donnée à la partie type du bus commun BUSA, et fournit les signaux dnp, dl, de, ei et maj. Une mémoire vive RAMFG de largeur 2 bits (dénommés respectivement ro et rw) reçoit sur son bus adresse les champs tag et cadre du bus commun BUSA. Le bus de donnée de cette mémoire, constitué des bits ro et rw, est relié d'une part à une logique PAL80, d'autre part à une porte logique ET80, en direct pour rw, de façon inversée pour ro. La logique PAL80 est reliée au champ type et reçoit des signaux logiques cl, r/w, s/n, mff et en82 : le signal cl est issu d'une bascule B82, les signaux r/w et s/n d'une file d'attente associative AFIFO, le signal mff d'une porte logique ET81, et le signal en82 d'une bascule B83, qui reçoit sur ses entrées S et R respectivement des signaux Q82 et Q81 inversés du distributeur de phases DP_M. La logique PAL câble sur ses sorties ro-rw les équations logiques suivantes : $\overline{dnp} = 10$; $\overline{dl.mff} = 10$; $\overline{dl.mff} = 01$; $\overline{de} = 01$; $\overline{maj.cl} = 00$; $\overline{ei} = 01$; $\overline{maj.cl.s/\overline{n}.r/w} = 10$; $\overline{maj.cl.s/\overline{n}.r/w} = 01$. La sortie de la porte logique ET80 est reliée à l'entrée D d'une bascule B84, qui reçoit sur son entrée horloge la phase Q82. La sortie Q de cette bascule est reliée à l'une des entrées de la porte logique ET81, qui reçoit sur son autre entrée la sortie de la porte logique OU80. Les deux entrées de cette porte logique OU80 sont reliées aux sorties de et et dl du décodeur DEC80. L'entrée de lecture r de la mémoire RAMFG est reliée à la phase Q81, et l'entrée d'écriture w à la sortie d'une porte logique ET82. La porte logique ET82 reçoit sur ses entrées le signal Q83 et la sortie d'une porte logique ET83, dont les entrées sont reliées au signal s/ \overline{n} et à la phase Q87. La

sortie dnp du décodeur DEC80 est reliée à des portes logiques ET84 et ET85, qui reçoivent sur leur autre entrée respectivement les phases 081 et 085. Le signal s/\bar{n} est également délivré vers des portes logiques ET86 et ET87 qui reçoivent sur leur autre entrée respectivement les phases 086 et 090. La sortie mff de la porte logique ET81 est également reliée à une porte logique ET88, qui reçoit sur son autre entrée la phase 083, et de façon inversée à des portes logiques ET89 et ET90 qui reçoivent sur leur autre entrée respectivement les phases 083 et 087. La sortie de la porte logique ET88 est reliée à l'entrée wff de la file AFIFO. Les sorties des portes logiques ET84, ET86, ET89 sont reliées aux entrées d'une porte logique OU81, dont la sortie est reliée à l'entrée S d'une bascule B85. Les sorties des portes logiques ET85, ET87, ET90 sont reliées aux entrées d'une porte logique OU82, dont la sortie est reliée à l'entrée R de la bascule B85. Le signal s/\bar{n} est également relié, de façon inversée, à une porte logique ET91 qui reçoit sur son autre entrée la phase 089. La sortie de la porte logique ET91 est reliée à l'entrée d'une porte logique NOU80 qui reçoit sur son autre entrée la sortie de la porte logique OU82. La sortie de la porte logique NOU80 est reliée à l'entrée \bar{R} de la bascule B80. La sortie maj du décodeur DEC80 est reliée à l'entrée de portes logiques ET92, ET93, ET94, ET95, qui reçoivent sur leur autre entrée respectivement les phases 081, 085, 085, 091. Les sorties des portes logiques ET92 et ET93 inversées sont reliées respectivement aux entrées S et R d'une bascule B86, et celles des portes logiques ET94 et ET95 aux entrées S et R d'une bascule B82. La sortie Q de la bascule B82 produit le signal logique cl également délivré à l'entrée cff de la file AFIFO et à l'entrée de commande sel80 d'un multiplexeur MUX80. La partie tag, cadre du bus commun BUSA est reliée à l'entrée de donnée de la file AFIFO et à l'une des entrées de données du multiplexeur MUX80. La sortie dl du décodeur DEC80 est également reliée à l'une des entrées de données de la file AFIFO afin de produire le signal de lecture/écriture l/e. La sortie de donnée de la file AFIFO est reliée à l'autre entrée du multiplexeur MUX80. La sortie du multiplexeur MUX80 est reliée au bus adresse de la mémoire centrale RAM pour la

partie tag.cadre, et aux entrées de décodeurs DEC81 et DEC82 pour la partie champ cpu. La sortie \overline{Q} de la bascule B86 est reliée à l'entrée d'écriture de la mémoire centrale RAM et à 5 l'entrée $\overline{en84}$ du décodeur DEC82. La sortie Q de la bascule B85, légèrement retardée, est reliée à l'entrée de lecture de la mémoire centrale RAM et à l'une des entrées d'une porte logique ET96, qui reçoit sur son autre entrée la sortie de la porte logique OU82. La sortie de la porte logique ET96 est 10 reliée à l'entrée en83 du décodeur DEC81. La sortie j du décodeur DEC81 est reliée à l'entrée de validation des buffers de passage du registre à décalage mémoire RDM_j et la sortie j du décodeur DEC82 à l'entrée de chargement dudit registre à décalage mémoire RDM_j.

15 Le fonctionnement de cet ensemble est le suivant :

L'activation du signal valid provoque le déclenchement du distributeur de phases DP_M, et la validation du décodeur DEC80, qui va permettre de déterminer la nature de 20 la requête. La phase 081 est utilisée pour lire l'état des bits correspondant au bloc demandé dans la mémoire RAMFG, et la combinaison ro.rw est mémorisée dans la bascule B84. Une première écriture a lieu dans la mémoire RAMFG sur la phase 083, qui permet de mettre à jour les bits d'états. Leur valeur 25 est fournie par la logique PAL80 et permet d'obtenir les enchaînements suivants :

- en cas de requête de bloc de données non partagées (dnp) alors quel que soit l'état des bits ro.rw (rw est forcément à zéro), l'état 10 est forcé ("bloc diffusé en 30 lecture") ;

- en cas de demande de bloc en lecture (dl) ou en écriture (de), si ro.rw = 01, alors la requête est mise 3 en file d'attente sur la phase 083 et l'état 01 est forcé (en fait, c'est l'état précédent), sinon l'état 10 est forcé en 35 cas de lecture ("bloc diffusé en lecture") et l'état 01 est forcé en cas d'écriture ("bloc diffusé en écriture") ;

- en cas de demande de mise à jour (maj), l'état 00 est forcé ("bloc non diffusé"). Dans ces divers cas, une lecture ou une écriture en mémoire centrale RAM est 40 opérée, vers ou à partir du registre à décalage mémoire RDM_j

identifié par le champ cpu du bus commun BUSA. Dans l'exemple choisi, la durée du cycle mémoire RAM est de 4 périodes de l'horloge générale h. En cas de lecture de données non 5 partagées, le cycle s'effectue de 081 à 085, dans les autres cas de 083 à 087. L'écriture s'effectue de 081 à 085 ;

- en cas d'écriture informative, celle-ci ne provoque pas de mouvement de données, mais force les bits d'états à la valeur 01 (l'état de départ est dans ce cas 10 forcément 10) ;

- en cas de demande de mise à jour, une consultation de la file d'attente AFIFO est systématiquement opérée. Cette consultation peut entraîner la lecture d'un bloc, dans le cas où une unité centrale CPU est en attente 15 dans la file AFIFO de mise à jour de ce bloc.

La lecture s'effectue de 086 à 090 et l'état des bits est forcé à 10 (demande pour une lecture) ou 01 (demande pour une écriture). La fin de toute opération se traduit par la remise à zéro de la bascule 080 qui active le 20 signal done. Cette désactivation peut se produire sur les phases 085, 087 ou 091 selon l'opération demandée, ou sur 089 si la consultation de la file donne un résultat négatif.

La file d'attente associative n'est pas détaillée. Elle est constituée de façon classique d'une 25 mémoire associative utilisée en file d'attente. Le nombre de mots de cette mémoire est égal au nombre d'unité centrale CPU du système multiprocesseur. Un "daisy-chain" interne identifie sur chaque phase 081 le prochain mot candidat à une écriture, qui se produit le cas échéant sur la phase 083 par le signal 30 wff. Le signal cff déclenche une comparaison à partir de la phase 085, les bascules de la mémoire de réponse ayant été remises à zéro sur la phase 084. Le résultat de la comparaison est répercuté sur le signal s/\bar{n} (some/ \bar{n} one) et le contenu du mot concerné est disponible sur la sortie de données à partir 35 de la phase 086. Ce mot est ensuite invalidé sur la phase 090.

Dans l'architecture ci-dessus décrite, les processeurs espions PE_j sont sollicités à chaque transfert d'adresses sur le bus commun BUSA, avec consultation éventuelle de leur mémoire-cache MC_j . Cette consultation est 40 la plupart du temps inutile (faible probabilité de présence de

l'adresse du bloc correspondant au transfert, dans les mémoires-caches).

Il est à noter que le processeur de gestion mémoire PGM maintient des bits d'état des blocs et rend possible une gestion centralisée du maintien de cohérence. A cet effet, l'on peut rajouter à l'architecture ci-dessus décrite (figure 10) un bus parallèle de synchronisation fonctionnant selon le même algorithme que le bus de synchronisation SYNCHRO de la variante qui est décrite ci-après. Les processeurs-espions ne sont plus alors à proprement parler des espions (puisque connectés sur le bus de synchronisation et non sur le bus commun BUSA), et sont désignés par processeurs de maintien de cohérence (PMC_j pour la variante de la figure 18). Ainsi, le processeur de gestion mémoire PGM demeure sollicité à chaque transfert sur le bus commun BUSA, mais les processeurs de maintien de cohérence sont sollicités par le processeur PGM uniquement lorsqu'ils sont concernés par le transfert.

La figure 18 présente une variante dans laquelle la cohérence est maintenue selon le principe ci-dessus évoqué. Cette variante reprend l'architecture générale de la figure 6, avec des adresses de blocs qui transitent par les liaisons séries LS_j. Ce système comprend un bus parallèle de synchronisation SYNCHRO, de même structure logique que le bus commun BUSA, mais piloté à la seule initiative du processeur de gestion mémoire PGM.

La structure de l'unité centrale UC_j est conforme à celle présentée à la figure 10, avec quelques modifications :

- la structure de la mémoire-cache MC_j reste la même, ainsi que la structure du répertoire de gestion RG_j,
- le processeur de gestion parallèle PGP_j disparaît, puisque le bus commun BUSA n'existe plus et les fonctions qui lui étaient attribuées sont reportées sur le processeur de gestion de la liaison série PGS_j ;
- le processeur espion PE_j est remplacé par un processeur de maintien de cohérence PMC_j qui s'occupe du maintien des bits d'états des blocs dans la mémoire-cache MC_j en vue d'en assurer la cohérence et qui est activé à la seule

67

initiative du processeur de gestion mémoire PGM_j via le bus de synchronisation SYNCHRO ;

- le processeur de gestion des requêtes PGU ne connaît plus qu'un seul partenaire : le processeur de gestion de la liaison série PGS_j, sur lequel il reporte toutes ses requêtes ;

- le processeur de gestion de la liaison série PGS_j est chargé du transfert des adresses et des données, conformément au principe décrit pour le système de la figure 6, chaque adresse étant préfixée par la nature de la requête ;

- les fonctionnalités du processeur de gestion mémoire PGM sont celles décrites en référence à la figure 17, son activation n'étant plus assurée par le signal valid, qui disparaît (puisqu'auparavant associé au bus commun BUSA), mais par l'arbitre ABM décrit dans le système de la figure 6, lequel sérialise les demandes de service qui transitent par les liaisons séries. La mémoire RAMFG est également constituée d'un champ supplémentaire cpu associé aux bits d'états ro.rw.

Le fonctionnement général du mode de réalisation représenté à la figure 18 est le suivant :

Chaque requête du processeur de traitement CPU active le processeur de gestion des requêtes PGU_j avec l'indication lecture ou écriture et code ou données. Ce processeur requiert un accès au répertoire de gestion RG_j auprès du processeur de gestion du répertoire PGR_j. La consultation du répertoire conduit à l'un des cas suivants :

- le bloc est présent dans la mémoire-cache MC_j, avec l'état non modifié ($m = 0$) ; si la demande est une lecture, l'information demandée est extraite de la mémoire-cache MC_j et fournie au processeur de traitement CPU_j. Si la demande est une écriture, alors une requête d'écriture informative ei est transmise au processeur de gestion de la liaison série PGS_j ;

- le bloc est présent dans la mémoire-cache MC_j, avec l'état modifié ($m = 1$) ; la demande, lecture ou écriture, est satisfaite ;

- le bloc est absent de la mémoire-cache

MC_j ; une requête de bloc en lecture ou écriture est transmise au processeur de gestion de la liaison série PGS_j.

Ainsi, les requêtes faites au processeur de gestion de la liaison série peuvent être : une demande de lecture de données non partagées (code) : dnp, une demande de lecture de bloc : dl, une demande de lecture de bloc en vue d'y effectuer une écriture : de, une demande d'écriture informative : ei.

10 A ces divers états, il faut ajouter l'état mise à jour maj correspondant à la purge d'un bloc, soit à la demande du processeur de maintien de cohérence PMC_j, soit pour libérer un emplacement de bloc dans la mémoire-cache. Les adresses ainsi préfixées transitent par les liaisons séries 15 LS_j, et conformément au principe énoncé lors de la description de l'architecture de la figure 6, sollicitent l'arbitre ABM lorsque :

- en cas de lecture bloc, l'adresse est transmise,
- 20 - en cas d'écriture bloc, l'adresse et la donnée sont transmises.

Ces requêtes sont traitées séquentiellement par le processeur de gestion mémoire PGM, de même structure générale que celle décrite à la figure 17. Leur traitement est 25 le suivant :

1/ dnp : demande de données non partagées. Le bloc est transmis et prend l'état ro.rw = 10.

2/ dl : demande de lecture d'un bloc.

Si le bloc est dans l'état "non diffusé" 30 (ro.rw = 00) ou "diffusé en lecture" (ro.rw = 10), il est transmis et prend ou garde l'état ro.rw = 01.

Si le bloc est dans l'état "diffusé en écriture", la demande est mise dans la file d'attente AFIFO. Le processeur de gestion mémoire PGM trouve alors dans le 35 champ cpu de la mémoire RAMFG l'adresse de la mémoire-cache MC_i qui contient la version à jour du bloc demandé. Une requête de purge est alors émise sur le bus de synchronisation SYNCHRO, à seule destination du processeur de maintien de cohérence PMC_i associé à la mémoire-cache MC_i concernée. Cette 40 demande peut être qualifiée de commande adressée.

On notera que le processeur de maintien de cohérence PMC_i n'a pas à consulter le répertoire de gestion RG_i associé puisque le processeur de gestion mémoire PGM a connaissance du fait qu'il est seul possesseur de la copie à jour. Son rôle consiste simplement à prélever la requête et à la déposer dans la file $FIFO_i$ associée.

3/ de : demande de lecture d'un bloc en vue d'y faire une écriture.

10 Si le bloc est dans l'état "non diffusé" ($ro.rw = 00$), il est transmis et prend l'état "diffusé en écriture" ($ro.rw = 01$).

Si le bloc est dans l'état "diffusé en lecture" ($ro.rw = 10$), alors le processeur de gestion mémoire émet une commande universelle d'invalidation de bloc, puis transmet le bloc avec l'état "diffusé en écriture" ($ro.rw = 01$). La commande universelle provoque l'activation de tous les processeurs de maintien de cohérence PMC_j qui exécutent strictement les mêmes opérations que celles décrites pour le système de la figure 10.

Si le bloc est dans l'état "diffusé en écriture", la demande est mise en file d'attente AFIFO. Comme précédemment, le processeur de gestion mémoire PGM émet une commande adressée au seul possesseur de la copie à jour.

25 4/ maj : demande d'écriture d'un bloc par suite d'une purge.

L'algorithme de fonctionnement est dans ce cas strictement le même que celui décrit en référence à la figure 17 pour le processeur PGM.

30 Il est à noter que le problème de l'acquittement de l'écriture trouve naturellement sa solution dans ce mode de réalisation par une commande adressée d'acquittement.

5/ ei : écriture informative.

35 Ce cas est traité directement sur le bus commun BUSA dans l'architecture présentée à la figure 10. Dans le mode de réalisation visé ici, et afin de garantir la synchronisation, cette opération est prise en charge par le processeur de gestion mémoire PGM.

40 Si le bloc est dans l'état "diffusé en

lecture", alors une commande, à la fois universelle et adressée, est émise : adressée en ce sens que le processeur de maintien de cohérence PMC_j concerné note l'acquittement de la demande d'écriture informative et passe le bloc concerné dans l'état "modifié" dans le répertoire de gestion RG_j , universelle en ce sens que tous les autres processeurs PMC_i doivent invalider ce bloc dans leur répertoire.

Le bloc dans l'état "diffusé en écriture" indique qu'une demande d'écriture informative a été traitée sur ce même bloc pendant le temps d'attente de traitement de la requête. Dans ce cas, la demande d'écriture informative est transformée en demande d'écriture de, et suit le même traitement que dans le cas correspondant de l'écriture.

Le bus parallèle de synchronisation SYNCHRO est chargé de diffuser des adresses de bloc préfixées par un numéro de processeur et un type de demande, soit environ 30 à 40 bits selon les caractéristiques du multiprocesseur. Ces informations sont de plus transmises de façon unidirectionnelle. Leur transfert peut alors avantageusement là encore se faire par une liaison série. La cadence de transfert est moins critique que pour les blocs, et des solutions simplifiées peuvent être envisagées, par exemple par le biais de circuits "TAXI" fabriqués par la Société "A.M.D."

La figure 19 présente un schéma synoptique partiel d'une architecture conforme à l'invention, dans laquelle plusieurs unités centrales UC_k ... sont réunies en grappe ("cluster") et partagent une même liaison série LS_k . Dans ce but, un arbitre local ABL_k associé à la grappe est chargé d'arbitrer les conflits d'accès aux moyens de communication d'adresses de blocs, et partage avec le processeur mémoire PGM, aménagé à cet effet, un signal $busy_k$ indiquant en permanence l'état libre ou occupé de la liaison série LS_k . Des moyens de codage et de décodage d'un en-tête d'identification du processeur concerné à l'intérieur d'une grappe sont associés aux logiques d'émission et de réception des blocs de données.

Dans le cas où les moyens de communication d'adresses de blocs sont constitués par le bus commun BUSA, le fonctionnement est le suivant :

Si l'unité centrale CPU_{k+j} désire faire un transfert de bloc dans le sens mémoire centrale RAM vers mémoire-cache MC_{k+j} (cas dnp, dl, de) ou effectuer une écriture informative ei, alors elle requiert l'accès vers le bus commun BUSA à l'arbitre local ABL_k, qui répercute la demande, son tour venu, à l'arbitre AB. L'accord d'accès au bus commun BUSA est renvoyé sur l'unité centrale CPU_{k+j} et le transfert s'effectue de la façon décrite en référence à la figure 10. Tout bloc transmis dans le sens mémoire centrale RAM vers mémoire-cache MC_{k+j} doit alors être identifié, car l'ordre des demandes n'est pas respecté du fait de la possibilité de mise en attente dans la file AFIFO du processeur de gestion mémoire PGM. Si l'unité centrale CPU_{k+j} désire faire un transfert de bloc dans le sens mémoire-cache MC_{k+j} vers la mémoire centrale RAM (cas maj), alors elle requiert tout d'abord de l'arbitre local ABL_k l'accès à la liaison série LS_k. L'arbitre local ABL_k et le processeur de gestion mémoire PGM sont susceptibles tous deux de s'attribuer la liaison série LS_k : la contention est évitée en synchronisant la modification du signal busy_k avec le signal valid (le processeur de gestion mémoire PGM ne peut lancer ou relancer un transfert qu'au cours d'une transaction mémoire). L'accord d'occupation de la liaison série LS_k conduit l'unité centrale CPU_{k+j} à transférer son bloc d'information vers le registre à décalage mémoire RDM_k, puis à requérir de l'arbitre local ABL_k l'accès au bus commun BUSA afin d'y effectuer la demande de mise à jour, qui s'effectue selon l'algorithme décrit en référence à la figure 10. L'écriture de mise à jour peut entraîner une libération de bloc dans la file d'attente AFIFO du processeur de gestion mémoire PGM et requérir un registre à décalages RDM_j occupé. Dans ce cas, le transfert demandé est retardé et chaîné au transfert en cours.

Dans le cas où les moyens de communication d'adresses de blocs sont les liaisons séries elles-mêmes, le fonctionnement est alors identique au cas précédent pour ce qui est de la préemption de la liaison série, et identique pour l'algorithme général de fonctionnement à celui présenté en référence à la figure 17.

Par exemple, une demande de lecture de bloc

issue de l'unité centrale CPU_{k+j} nécessite tout d'abord un accord d'accès à la liaison série LS_k , accord donné par l'arbitre local ABL_k en concertation avec le processeur de gestion mémoire PGM. L'accord d'accès entraîne le transfert de l'adresse du bloc demandé sur la liaison série LS_k , qui est aussitôt libérée : elle est disponible pour toute autre transaction si nécessaire. Une demande d'écriture de bloc suit le même protocole pour l'accès à la liaison série LS_k .

10 Dans les architectures décrites en référence aux figures 1 à 19, il existait autant de registres à décalages mémoires RDM_j que d'unités centrales CPU_j : une liaison série LS_j était affectée de façon statique à un couple (RDM_j , CPU_j).

15 S'il doit à l'évidence exister au moins une liaison série LS_j entre une unité centrale et la mémoire centrale RAM, le nombre de registres à décalages RDM_j peut être inférieur. En effet, si t_{acc} est le temps d'accès à la mémoire centrale RAM et t_{tfr} le temps de transfert d'un bloc, 20 il n'est pas possible de maintenir plus de $n = t_{tfr}/t_{acc}$ registres à décalages simultanément occupés. Par exemple, pour $t_{acc} = 100$ ns et $t_{tfr} = 1\ 200$ ns, on obtient $n = 12$.

t_{acc} et t_{tfr} sont alors des critères caractéristiques de la performance du système multiprocesseur 25 conforme à l'invention et la mise en place de n registres à décalages mémoires RDM_j est compatible avec un nombre supérieur de liaisons séries LS_j qu'à la condition d'intercaler une logique de type réseau d'interconnexion RI entre registres et liaisons, l'affectation d'un registre 30 mémoire RDM_j à une liaison série LS_j étant réalisée de façon dynamique par le processeur de gestion mémoire PGM.

Par ailleurs, la mémoire centrale RAM sera généralement constituée par m bancs mémoires $RAM_1, \dots, RAM_p, RAM_m$ agencés en parallèles, chaque banc mémoire comportant n 35 registres à décalages RDM_j reliés par un réseau d'interconnexion RI_p à l'ensemble des liaisons séries LS_j . Il est alors possible, sous réserve que les adresses de blocs soient uniformément distribués sur les bancs mémoires RAM_p , d'obtenir une performance théorique de $m \times n$ registres à 40 décalages simultanément actifs. La distribution uniforme des

adresses est assurée par des mécanismes classiques d'entrelaçages des adresses.

Sur la figure 20a, on a représenté partiellement une architecture conforme à l'invention, comportant m bancs mémoires RAM_p avec n registres à décalages RDM_j par bancs mémoires et q unités centrales UC_j . Chaque banc mémoire est de type à accès aléatoire doté d'une entrée/sortie de données de largeur correspondant à un bloc d'informations bi , cette entrée/sortie étant (comme précédemment pour la mémoire RAM) reliée par un bus parallèle à l'ensemble des registres élémentaires $RDM_{1p} \dots RDM_{jp}$.

Le réseau d'interconnexion est de structure connue en soi ("cross-bar", "delta", "banyan"...). On notera qu'un réseau multi-étage est bien adapté dans la mesure où le temps d'établissement du chemin est négligeable devant son temps d'occupation (le temps de transfert d'un bloc) et qu'il ne concerne qu'un bit par liaison.

Le processeur de gestion mémoire PGM est adapté pour pouvoir assurer l'allocation dynamique d'une sortie à une entrée du réseau, c'est-à-dire mettre en relation un registre à décalage mémoire RDM_j et une liaison série LS_i .

Dans le cas où les moyens de communication d'adresses de blocs sont constituées par le bus commun BUSA, le fonctionnement est le suivant :

En cas de demande de lecture d'un bloc de la part de l'unité centrale CPU_j , le processeur de gestion mémoire PGM_p concerné alloue un registre à décalage RDM_i , commande le réseau d'interconnexion RI en conséquence et initialise le transfert.

En cas de demande d'écriture d'un bloc de la part de l'unité centrale CPU_j , un chemin doit au préalable être établi. A cet effet, une première requête d'établissement de chemin est émise sur le bus commun BUSA, suivie de la demande d'écriture effective dès transfert du bloc de la mémoire-cache MC_j au registre à décalage RDM_i . Lors de la première demande, le processeur de gestion mémoire PGM est chargé d'allouer un chemin et de commander le réseau d'interconnexion RI.

Dans le cas où les moyens de communication

d'adresses de blocs sont les liaisons séries elles-mêmes, un chemin doit être établi à tout préalable de transfert. Ce problème est identique au problème classique du partage d'un ensemble de n ressources par m utilisateurs et peut être réglé par des solutions classiques d'arbitrages de conflits d'accès (protocoles de communication, signaux supplémentaires).

Dans l'exemple d'architecture représenté à la figure 5, les registres à décalages RDM_j et RDP_j , leurs logiques de validation $LV1$ et $LV2$ étaient réalisées dans une technologie rapide, l'ensemble étant synchronisé par une horloge de fréquence F au moins égale à 100 MHz.

La figure 20b présente en variante à l'architecture proposée à la figure 20a, une solution conforme à l'invention dans laquelle chaque liaison série LS_j , qui relie le processeur CPU_j à tous les bancs mémoires, est éclatée en m liaisons séries LS_{jp} reliant en point en point le processeur CPU_j à chacun des bancs mémoires RAM_p .

Ce procédé présente le double avantage suivant :

- chaque liaison étant du type point à point, peut mieux être adaptée du point de vue électrique ou du point de vue fibre optique,
- un niveau de parallélisme supplémentaire est obtenu dès lors que le processeur de traitement est en mesure d'anticiper des demandes de blocs, ce qui est le cas actuellement pour les processeurs les plus performants.

La logique d'interface (notée précédemment TFR_j et RDP_j) qui est associée à la liaison série LS_j , côté processeur CPU_j , est alors dupliquée en m exemplaires $I_1... I_p... I_m$. On notera la présence d'une liaison de maintien de cohérence de l'information, privée à chaque banc mémoire RAM_p . Le fonctionnement de cette liaison est analogue à celui du bus SYNCHRO de la figure 18.

On a représenté aux figures 21a et 21b une autre structure de mémoire RAM qui comprend 2^u plans mémoires, chaque plan mémoire présentant un front de $t/2^u$ informations binaires (pour des raisons de clarté, l'on a représenté à la figure 21a les moyens nécessaires à la lecture d'un bloc bi , et à la figure 21b les moyens nécessaires à l'écriture). Les

registres à décalages RDM_j ou RDP_j sont constitués de 2^u sous-registres à décalages élémentaires RDM_{jp} à $t/2^u$ bits de capacité. L'exemple présenté à la figure 20 est une réalisation à 8 plans mémoires ($u = 3$). (Pour la clarté du dessin, il a été représenté un seul registre à décalage RDM_j formé pour l'ensemble des sous-registres RDM_{jp}). Chaque plan mémoire RAM_p comporte en regard de son front d'accès un ensemble de registres à décalages élémentaires RDM_{jp} et est capable de fonctionner avec une fréquence de décalage d'au moins $F/2^u$.

Le fonctionnement de l'ensemble en cas de lecture est illustré à la figure 21a. Un bloc est lu de façon synchrone dans l'ensemble des 2^u plans mémoires et chargé de même dans les registres élémentaires de même rang. Les sorties séries de ces registres sont connectées aux entrées d'un multiplexeur MUXR réalisé en technologie rapide (ASGA). Un circuit de ce type, parfaitement adapté, est disponible chez "GIGABIT LOGIC", sous la référence "10G040", et est capable de délivrer un signal logique à une fréquence de 2,7 GHz. Il fournit par ailleurs une horloge de fréquence divisée par huit, qui constitue l'horloge de décalage des registres élémentaires RDM_{jp} .

En cas d'écriture, un fonctionnement symétrique, présenté à la figure 21b, est obtenu avec un circuit de multiplexeur DMUXR du même fabricant (référéncé "10G41"), avec les mêmes caractéristiques de performances.

On obtient ainsi une fréquence de transfert de 500 MHz avec 8 registres élémentaires fonctionnant à une fréquence de $500/8 = 62,5$ MHz, ce qui les rend réalisables dans une technologie plus conventionnelle ("MOS" par exemple).

Les circuits multiplexeurs et démultiplexeurs référencés ci-dessus sont combinables en ensembles de 16, 32, ... bits. Ainsi, en associant respectivement 16, 32 plans mémoires fonctionnant à 62,5 MHz, il est possible d'obtenir des débits de 1 et 2 GHz, soit un niveau de performances 2 à 4 fois supérieur.

Il est à noter que la logique TFR peut être réalisée sur l'un des registres élémentaires, et que la logique de validation LV est intégrée aux circuits "ASGA"

(sortie à collecteur ouvert).

La figure 22 présente la structure générale d'un composant de type circuit intégré "VLSI", dénommé 5 "mémoire multiport série" et susceptible d'équiper un système multiprocesseur conforme à l'invention. Ce composant peut être utilisé dans l'architecture multi-processeurs décrite précédemment, soit pour réaliser la mémoire centrale RAM et les registres à décalage RDM_j associés, soit pour réaliser 10 chaque mémoire-cache MC_j et son registre à décalage RDP_j. Pour simplifier les notations, on a gardé dans la description qui suit les symboles relatifs à la mémoire centrale RAM et les registres à décalage associés.

La liste des broches de ce circuit avec les 15 signaux correspondants est la suivante :

- $adbloc_0$ - $adbloc_{m-1}$: m bits d'adresses de bloc bi,
- $admot_0$ - $admot_{k-1}$: k bits d'adresses de mots dans le bloc,
- 20 - $numreg_0$ - $numreg_{n-1}$: n bits d'adresses de registres rd,
- \overline{cs} : "chip select" : signal de sélection du circuit,
- \overline{wr} : "write" : signal d'écriture,
- 25 - \overline{rd} : "read" : signal de lecture,
- $\overline{bit/bloc}$: signal de commande de la fonction multiport,
- $\overline{normal/config}$: signal de mode de fonctionnement,
- 30 - $data_0$ - $data_{l-1}$: l bits de données,
- h_1 - h_n : n signaux d'horloge,
- d_1 - d_n : n signaux de données.

Les valeurs m, n, l sont fonctions de l'état 35 courant de technologie. Des valeurs actuelles pourraient être les suivantes :

- m = 16 soit 2^{16} blocs bi de 64 bits chacun (soit 4 Mbits),
- n = 3 soit 8 registres rd,
- l = 8 soit un interface parallèle de type 40 octets,

- $k = 3$ du fait de la présence de 8 octets par bloc.

Le composant visé comporte une cinquantaine de broches.

Ce circuit mémoire multiport série est composé d'une mémoire vive à accès aléatoire RAM, de largeur prédéterminée t , susceptible d'être commandée en écriture sur des fronts indépendants de largeur $t/4$ (valeur choisie à titre d'exemple sur la figure 22) et $t/1$. Les lignes de données de cette mémoire RAM sont reliées aux entrées d'une logique de type "barillet" BS ("Barrel shifter"), ou de multiplexage MT, selon la version de composant, la logique de multiplexage MT pouvant être considérée comme offrant un sous-ensemble des possibilités de la logique "barillet" et donc plus simple à réaliser. Les signaux d'adresses et de commande de cette mémoire RAM, à savoir csi, wri, rdi, adblocki, sont délivrés à partir d'une logique de commande COM. Cette logique COM reçoit en outre les signaux d'informations des broches \overline{cs} , \overline{wr} , \overline{rd} , bit/bloc, normal/config, numreq et est reliée, d'une part, par des lignes de commande "format" à la logique de type "barillet" BS, d'autre part, à la sortie d'un registre de configuration RC1, et à l'entrée d'une logique de sélection LSR fournissant des signaux src_0, \dots, src_{n-1} et src_1, src_2, src_3 . Les sorties de la logique de type barillet BS constituent un bus interne de communication parallèle BUSI, relié à un ensemble de registres à décalages RD_0, \dots, RD_{n-1} , d'une part, sur leurs entrées parallèles et, d'autre part, sur leurs sorties parallèles à travers des buffers de validation $BV100_0, \dots, BV100_{n-1}$ et à l'entrée parallèle des registres de configuration RC_1, RC_2, \dots, RC_i .

Les 1 bits de faible poids du bus BUSI sont également reçus sur les 1 broches $data_0, \dots, data_{1-1}$. Chaque registre à décalage RD_i et des portes logiques associées constituent une unité fonctionnelle $ELRD_i$, pilotée par un ensemble d'éléments logiques qui constituent une logique de forçage LF_i . Chaque unité fonctionnelle $ELRD_i$ comporte des portes logiques $ET100_i$ et $ET101_i$, reliées sur l'une de leur entrée à la sortie src_i de la logique de sélection LSR, et recevant sur leur autre entrée respectivement les signaux rd_i

et wr_i . La sortie de la porte logique $ET100_i$ est reliée à l'entrée $load100_i$ du registre à décalage RD_i , ainsi qu'à l'entrée $load101_i$ et à l'entrée S respectivement d'un compteur 5 $CPT100_i$ et d'une bascule $B100_i$ appartenant à la logique de forçage LF_i . La sortie de la porte logique $ET101_i$ est reliée à l'entrée de commande des buffers de validation $BV100_i$. La sortie di est connectée à la sortie d'une porte logique PL_i , qui reçoit sur son entrée de donnée la sortie série du 10 registre à décalage RD_i et sur son entrée de commande la sortie d'une porte logique $OU100_i$. Le signal issu de la broche h_i est délivré à l'entrée $clk100_i$ du registre RD_i ainsi qu'à l'entrée $down100_i$ du compteur $CPT100_i$. La sortie $zérol00_i$ du compteur $CPT100_i$ est reliée à l'entrée R de la bascule $B100_i$.

15 La logique de forçage LF_i comporte en outre un multiplexeur $MUX100_i$ qui reçoit sur ses entrées de données les valeurs t et $t/4$. La sortie de donnée du multiplexeur $MUX100_i$ est reliée à l'entrée de donnée du compteur $CPT100_i$, et la commande de sélection $sell00_i$ du multiplexeur $MUX100_i$ 20 est reliée à la sortie l du registre RC_1 . La sortie Q de la bascule $B100_i$ est reliée à l'une des entrées d'une porte logique $ET102_i$, qui reçoit sur son autre entrée le signal issu de la broche i d'un registre RC_2 . La sortie de la porte logique $ET102_i$ est reliée à l'une des entrées de la porte 25 logique $OU100_i$, qui reçoit sur son autre entrée le signal issu de la broche i d'un registre RC_3 . Les entrées de chargement des registres RC_1 , RC_2 , RC_3 reçoivent respectivement les signaux src_1 , src_2 , src_3 issus de la logique de sélection LSR.

Ce composant présente une dualité de 30 fonctionnement : si le signal bit/\overline{bloc} est dans l'état "bit", alors le fonctionnement de ce composant est celui d'une mémoire à semi-conducteur conventionnelle : les signaux $adblock$ associés aux signaux $admot$ constituant le bus d'adresse en unité mot (8 bits dans l'exemple), les signaux \overline{cs} , \overline{rd} , \overline{wr} ont 35 le sens habituel attribué à ces signaux, et les broches data véhiculent les données.

De façon interne, en lecture le bloc d'information désigné par $adblock$ est lu en mémoire RAM et présenté à l'entrée de la logique de barillet BS ou de 40 multiplexage MT. La combinaison des signaux $admot$ et bit/\overline{bloc}

permettent à la logique de commande COM de fournir à la logique de barillet BS ou de multiplexage MT les signaux "format". Le mot concerné est alors cadré à droite en sortie de la logique de barillet ou de multiplexage et présenté ainsi sur les broches de donnée data.

De façon interne, en écriture le mot présenté sur les lignes de donnée data est cadré par la logique de barillet LS ou de multiplexage MT par les mêmes signaux de commande format qu'en lecture, en regard de sa position dans le bloc. La logique de commande COM émet alors un signal d'écriture wri partiel sur le seul "tronçon" de mémoire concerné, et à l'adresse désignée par les signaux adbloc.

Si le signal bit/bloc est dans l'état "bloc", alors le fonctionnement dépend de l'état du signal normal/config. Le mode config programme les registres de configuration RC_1 , RC_2 , RC_3 adressés par les signaux numreg, et programmés à partir des lignes de données data. Le registre RC_1 permet de modifier la taille du bloc : t et t/4 dans l'exemple, soit 64 bits et 16 bits. De façon interne, le fonctionnement est similaire à celui décrit dans le mode de fonctionnement "bit" : t ou t/4 bits sont cadrés sur le bus interne BUSI (en lecture), ou en regard du "tronçon" de bloc concerné (en écriture). Des tailles de blocs multiples peuvent être envisagées (t, t/2, t/4...).

Le registre RC_3 permet de choisir pour chaque registre un sens permanent de fonctionnement : soit en entrée ($RC3_i = 0$) soit en sortie ($RC3_i = 1$). Ce sens permanent permet d'adapter le composant aux liaisons séries à liens unidirectionnels permanents. Le registre RC_2 permet de choisir pour chaque registre, sous réserve que le bit correspondant de RC_3 soit à l'état logique 0, un mode de fonctionnement à liens bidirectionnels alternés : sur une lecture mémoire RAM, le registre à décalage RD_i concerné "passe" en mode sortie pour le temps de la transmission du bloc, puis revient à l'état de repos en mode "entrée". De façon interne, la bascule $B100_i$, qui pilote la porte logique PL_i , est mise à 1 sur un signal de chargement du registre RDM_i et remise à zéro à l'issue du transfert des t ou t/4 bits, par l'intermédiaire du compteur $CPT100_i$, initialisé à t ou t/4 selon l'état du registre RC_1 ,

et qui reçoit sur son entrée de décomptage les impulsions d'horloge hi. En fonctionnement normal (signal normal/config dans l'état normal) pour une lecture, le bloc adressé par les 5 broches adbloc est chargé dans le registre RD_i adressé par les broches numreg. Si le bloc est partiel (t/4), alors il est transmis en position faible poids sur le bus interne BUSI par la logique de type barillet BS ou de multiplexage MT. Ce bloc est alors transmis dès activation du signal d'horloge hi.

10 En fonctionnement normal pour une écriture, le contenu du registre RD_i adressé par les broches numreg est écrit dans le bloc mémoire RAM d'adresse adbloc. Si le bloc est partiel, il est transmis en position fort poids sur le bus interne BUSI, puis cadré en "regard" du tronçon de bloc 15 concerné par la logique de type barillet BS ou de multiplexage MT, et enfin un signal partiel d'écriture wri est émis sur le tronçon concerné.

Il est à noter que si un bloc partiel est en service, alors l'adresse de ce bloc partiel dans le bloc est 20 fournie par les lignes d'adresse admot.

Ce composant est parfaitement adapté aux diverses variantes d'architectures décrites. Associés en parallèle, 8, 16... circuits de ce type permettent de réaliser le dispositif décrit aux figures 20a, 20b. Si la mémoire RAM 25 est en technologie rapide, alors ce composant peut également être utilisé au niveau de la mémoire-cache, en multiplexant, selon le dispositif décrit aux figures 20a, 20b, les registres internes d'un même composant.

REVENDEICATIONS

1/ - Système multiprocesseur, du type comprenant une mémoire centrale (RAM) organisée en blocs d'informations (bi), des processeurs de traitement (CPU₁... CPU_j... CPU_n), une mémoire-cache (MC_j) reliée à chaque processeur de traitement (CPU_j) et organisée en blocs d'informations (bi) de même taille que ceux de la mémoire centrale, un répertoire (RG_j) et son processeur de gestion (PG_j) associé à chaque mémoire-cache (MC_j), des moyens de communication d'adresses de blocs entre processeurs (CPU_j) et mémoire centrale (RAM), ledit système multiprocesseur étant caractérisé en ce qu'il est doté :

. d'un ensemble de registres à décalage, dit registres-mémoire (RDM₁... RDM_j... RDM_n), chaque registre (RDM_j) de cet ensemble étant connecté à la mémoire centrale (RAM) de façon à permettre, en un cycle de cette mémoire, un transfert parallèle en lecture ou écriture d'un bloc d'informations (bi) entre ledit registre et ladite mémoire centrale,

. des registres à décalage, dits registres-processeur (RDP₁... RDP_j... RDP_n), chaque registre à décalage processeur (RDP_j) étant relié à la mémoire-cache (MC_j) d'un processeur (CPU_j) de façon à permettre un transfert parallèle en lecture ou écriture d'un bloc d'informations (bi) entre ledit registre à décalage (RDP_j) et ladite mémoire-cache (MC_j),

. un ensemble de liaisons séries (LS₁... LS_j... LS_n), chacune reliant un registre à décalage mémoire (RDM_j) et un registre à décalage processeur (RDP_j) et adapté pour permettre le transfert de blocs d'informations (bi) entre les deux registres considérés (RDM_j, RDP_j).

2/ - Système multiprocesseur selon la revendication 1, caractérisé en ce que :

- chaque registre à décalage mémoire (RDM_j) et chaque registre à décalage processeur (RDP_j) sont dédoublés en deux registres, l'un spécialisé pour le transfert dans un sens, l'autre pour le transfert dans l'autre sens,

- chaque liaison série (LS_j) comprend deux liens séries unidirectionnels de transfert bit à bit, reliant

le registre à décalage mémoire (RDM_j) dédoublé et le registre à décalage processeur correspondant (RDP_j) dédoublé, ces liens étant connectés auxdits registres pour permettre, l'un un transfert dans un sens, l'autre un transfert dans l'autre sens.

3/ - Système multiprocesseur selon la revendication 1, caractérisé en ce que chaque liaison série (LS_j) comprend un lien bidirectionnel de transfert bit à 10 bit, connecté au registre à décalage mémoire (RDM_j) et au registre à décalage processeur correspondant (RDP_j) et une logique (LV) de validation du sens de transfert de façon à permettre un transfert alterné dans les deux sens.

4/ - Système multiprocesseur selon l'une des revendications 1, 2 ou 3, caractérisé en ce que les moyens de communication d'adresses comprennent un bus commun de communication parallèle d'adresses de blocs (BUSA) reliant les processeurs (CPU_j) et la mémoire centrale (RAM) et un arbitre de bus (AB) adapté pour gérer les conflits d'accès audit bus.

5/ - Système multiprocesseur selon l'une des revendications 1, 2 ou 3, caractérisé en ce que les moyens de communication d'adresses comprennent un registre à décalage complémentaire (RDC_j) connecté sur chaque liaison série (LS_j) en parallèle avec le registre à décalage mémoire correspondant (RDM_j) de façon à permettre la transmission des adresses par les liaisons séries et leur chargement dans lesdits registres à décalage complémentaires (RDC_j), un arbitre de gestion d'accès (ABM) étant relié auxdits registres à décalage complémentaires (RDC_j) et à la mémoire centrale (RAM) en vue de prélever les adresses contenues dans lesdits registres (RDC_j) et de gérer les conflits d'accès à la mémoire centrale (RAM).

6/ - Système multiprocesseur selon l'une des revendications 1, 2, 3, 4 ou 5, comprenant des moyens de gestion des données partagées entre processeurs, en vue d'en assurer la cohérence.

7/ - Système multiprocesseur selon la revendication 6, caractérisé en ce que les moyens de gestion des données partagées comprennent :

• un bus spécial de communication parallèle

de mots (BUSD) reliant les processeurs (CPU_j) et la mémoire centrale (RAM),

. une logique de partition (LP_j), associée à
5 chaque processeur (CPU_j) et adaptée pour différencier les
adresses des données partagées et celles des données non
partagées de façon à transmettre celles-ci sur les moyens de
communication d'adresses avec leur identification,

. une logique de décodage (DEC) associée à la
10 mémoire centrale (RAM) et adaptée pour recevoir les adresses
avec leur identification et aiguiller les données en sortie
mémoire soit vers le registre à décalage mémoire correspondant
(RDM_j) pour les données non partagées, soit vers le bus
spécial de communication de mots (BUSD) pour les données
15 partagées.

8/ - Système multiprocesseur selon la
revendication 6, caractérisé en ce que les moyens de gestion
des données partagées comprennent, d'une part, un bus spécial
de communication parallèle de mots (BUSD) et un bus spécial
20 commun de communication d'adresses de mots (BUSAM) reliant les
processeurs (CPU_j) et la mémoire centrale (RAM), d'autre part,
une logique de partition (LP_j), associée à chaque processeur
(CPU_j) et adaptée pour différencier les adresses des données
partagées et celles des données non partagées, de façon à
25 aiguiller les premières vers le bus spécial commun (BUSAM) et
les secondes vers les moyens de communication d'adresses de
bloc.

9/ - Système multiprocesseur selon les
revendications 4 et 6 prises ensemble, caractérisé en ce que
30 les moyens de gestion des données partagées comprennent un
processeur de gestion mémoire (PGM) associé à la mémoire
centrale (RAM) et un processeur espion de bus (PE_j) associé à
chaque processeur de traitement (CPU_j) et au répertoire de
gestion correspondant (RG_j), chaque processeur espion de bus
35 (PE_j) et le processeur de gestion mémoire (PGM) étant
connectés au bus de communication d'adresses (BUSA) en vue
respectivement de surveiller et de traiter les adresses de
blocs transmises sur ledit bus de façon à permettre une mise à
jour de la mémoire centrale (RAM) et de la mémoire-cache
40 associée (MC_j) en cas de détection d'une adresse de bloc

présente dans le répertoire associé (RG_j).

10/ - Système multiprocesseur selon les revendications 4 et 6 prises ensemble, caractérisé en ce que
5 les moyens de gestion des données partagées comprennent un processeur de gestion mémoire (PGM) associé à la mémoire centrale (RAM) et un processeur de maintien de la cohérence des données partagées (PMC_j) associé à chaque processeur de traitement (CPU_j) et au répertoire de gestion correspondant
10 (RG_j), chaque processeur de maintien de cohérence (PMC_j) étant connecté à un bus de synchronisation (SYNCHRO) piloté par le processeur de gestion mémoire (PGM), de façon à permettre une mise à jour de la mémoire centrale (RAM) et de la mémoire-cache associée (MC_j) en cas de détection d'une adresse de
15 bloc, une mise à jour de la mémoire centrale (RAM) et des mémoires-caches (MC_j) à chaque prélèvement d'adresses sur le bus commun d'adresses BUSA.

11/ - Système multiprocesseur selon les revendications 5 et 6 prises ensemble, caractérisé en ce que
20 les moyens de gestion des données partagées comprennent un processeur de gestion mémoire (PGM) associé à la mémoire centrale (RAM) et un processeur de maintien de la cohérence des données partagées (PMC_j) associé à chaque processeur de traitement (CPU_j) et au répertoire de gestion correspondant
25 (RG_j), chaque processeur de maintien de cohérence (PMC_j) étant connecté à un bus de synchronisation (SYNCHRO) piloté par le processeur de gestion mémoire (PGM), de façon à permettre une mise à jour de la mémoire centrale (RAM) et de la mémoire-cache associée (MC_j) en cas de détection d'une adresse de
30 bloc, une mise à jour de la mémoire centrale (RAM) et des mémoires-caches (MC_j) à chaque prélèvement d'adresses dans les registres à décalage complémentaires (RDC_j).

12/ - Système multiprocesseur selon l'une des revendications 1 à 11, caractérisé en ce que :

35 - plusieurs registres à décalage processeur ($RDP_k, RDP_{k+1}...$) correspondant à un ensemble de processeurs déterminé ($CPU_k, CPU_{k+1}...$) sont connectés en parallèles à une même liaison série (LS_k), un arbitre local (ABL_k) étant associé à chaque ensemble de processeurs ($CPU_k, CPU_{k+1}...$) en
40 vue d'arbitrer les conflits d'accès à la liaison série (LS_k),

- un processeur de gestion mémoire (PGM) est relié aux moyens de communication d'adresses de blocs et à la mémoire centrale (RAM) et comprend des moyens de codage adaptés pour associer à chaque bloc d'informations (bi) un en-tête d'identification du processeur concerné parmi chaque ensemble (CPU_k , $CPU_{k+1}...$) partageant une liaison série donnée (LS_k),

10 - les processeurs de gestion (PG_k , $PG_{k+1}...$) associés aux mémoires-caches (MC_k , $MC_{k+1}...$) des processeurs de l'ensemble précité (CPU_k , $CPU_{k+1}...$) comprennent des moyens de décodage de l'en-tête d'identification.

13/ - Système multiprocesseur selon l'une des revendications 1 à 12, caractérisé en ce que chaque registre à 15 décalage mémoire (RDM_j) est connecté de façon statique à une liaison série (LS_j) spécifiquement affectée audit registre.

14/ - Système multiprocesseur selon l'une des revendications 1 à 12, caractérisé en ce que :

20 - un processeur de gestion mémoire (PGM) est associé à la mémoire centrale (RAM) et comprend une logique (ALLOC) d'affectation des registres à décalage mémoire aux liaisons séries,

- les registres à décalage mémoire ($RDM_1... RDM_j... RDM_n$) sont connectés de façon dynamique aux liaisons 25 séries ($LS_1... LS_j...$) par l'entremise d'un réseau d'interconnexion (RI) commandé par le processeur de gestion mémoire (PGM).

15/ - Système multiprocesseur selon l'une des revendications 1 à 14, dans lequel la mémoire centrale (RAM) 30 est constituée par m bancs mémoires ($RAM_1... RAM_p... RAM_m$) agencés en parallèle, caractérisé en ce que chaque registre à décalage mémoire (RDM_j) est constitué par m registres à décalage élémentaires ($RDM_{j1}... RDM_{jp}... RDM_{jm}$) reliés en parallèles à la liaison série correspondante (LS_j), chaque 35 registre élémentaire (RDM_{jp}) étant connecté à un banc mémoire (RAM_p) de façon à permettre, en un cycle dudit banc mémoire, un transfert parallèle en lecture ou écriture d'un bloc d'informations (bi) entre ledit registre élémentaire et ledit banc mémoire.

40

16/ - Système multiprocesseur conforme à la

revendication 15, dans lequel chaque liaison série (LS_j) est éclatée en m liaisons séries (LS_{jp}), reliant en point à point chaque processeur (CPU_j) au registre à décalage 5 élémentaire (RDM_{jp}).

17/ - Système multiprocesseur selon la revendication 15, dans lequel chaque banc mémoire (RAM_p) est du type à accès aléatoire doté d'une entrée/sortie de données de largeur correspondant à un bloc d'informations (bi), 10 caractérisé en ce que ladite entrée/sortie de chaque banc mémoire (RAM_p) est reliée par un bus parallèle à l'ensemble des registres élémentaires ($RDM_{1p}... RDM_{jp}$).

18/ - Système multiprocesseur selon l'une des revendications 15, 16 ou 17, synchronisé par une horloge de 15 fréquence F au moins égale à 100 mégahertz, caractérisé en ce que chaque registre à décalage élémentaire mémoire (RDM_{jp}) et chaque registre à décalage processeur (RDP_j) sont d'un type adapté pour présenter une fréquence de décalage au moins égale à F .

20 19/ - Système multiprocesseur selon l'une des revendications 15, 16 ou 17 synchronisé par une horloge de fréquence F au moins égale à 100 mégahertz, caractérisé en ce que chaque registre à décalage élémentaire mémoire et/ou chaque registre à décalage processeur est constitué d'un 25 ensemble de 2^u sous-registres multiplexés (RDM_{jp} , RDP_{jp}), chacun apte à présenter une fréquence de décalage au moins égale à $F/2^u$.

20/ - Procédé d'échange d'informations entre une mémoire centrale (RAM) organisée en blocs 30 d'informations (bi) et des processeurs ($CPU_1... CPU_j... CPU_n$) chacun doté d'une mémoire-cache (MC_j) organisée en blocs de même taille (bi), et d'un répertoire (RG_j) et de son processeur de gestion (PG_j), de façon que l'échange entre mémoire centrale (RAM) et chaque processeur (CPU_j) s'effectue 35 via la mémoire-cache (MC_j) de ce dernier, ledit procédé étant caractérisé en ce que chaque transfert de bloc d'informations (bi) depuis la mémoire centrale (RAM) vers la mémoire-cache (MC_j) d'un processeur donné (CPU_j) consiste :

. à transférer, en un cycle de mémoire 40 centrale, le bloc (bi) de ladite mémoire centrale (RAM) vers

un registre à décalage mémoire (RDM_j) de la taille d'un bloc, faisant partie d'un ensemble de registres à décalage ($RDM_1... RDM_j... RDM_n$) connectés à la mémoire centrale,

- 5 . à transférer sur une liaison série (LS_j) le contenu du registre à décalage mémoire (RDM_j) vers un registre à décalage processeur (RDP_j) de même capacité, associé à la mémoire-cache (MC_j) du processeur considéré (CPU_j),
10 . à transférer le contenu dudit registre à décalage processeur (RDP_j) vers ladite mémoire-cache (MC_j).

21/ - Procédé d'échange d'informations entre une mémoire centrale (RAM) organisée en blocs d'informations (bi) et des processeurs ($CPU_1... CPU_j... CPU_n$) chacun doté d'une mémoire-cache (MC_j) organisée en blocs de
15 mêmes tailles (bi), et d'un répertoire (RG_j) et de son processeur de gestion (PG_j), de façon que l'échange entre mémoire centrale (RAM) et chaque processeur (CPU_j) s'effectue via la mémoire-cache (MC_j) de ce dernier, ledit procédé étant caractérisé en ce que chaque transfert de bloc d'informations
20 (bi) depuis la mémoire-cache (MC_j) d'un processeur donné (CPU_j) vers la mémoire centrale (RAM) consiste :

- . à transférer le bloc (bi) de ladite mémoire-cache considérée (MC_j) vers un registre à décalage processeur (RDP_j) de la taille d'un bloc, associé à ladite
25 mémoire-cache (MC_j),

. à transférer sur une liaison série (LS_j) le contenu du registre à décalage processeur (RDP_j) vers un registre à décalage mémoire (RDM_j) de même capacité, affecté au processeur considéré dans un ensemble de registres à
30 décalage ($RDM_1... RDM_j... RDM_n$) connectés à la mémoire centrale (RAM),

. à transférer, en un cycle de mémoire centrale, le contenu du registre à décalage mémoire (RDM_j) vers ladite mémoire centrale (RAM).

- 35 22/ - Composant mémoire multiport série, susceptible d'équiper un système multiprocesseur conforme à l'une des revendications 1 à 19, caractérisé en ce qu'il est constitué par un circuit intégré comprenant une mémoire à accès aléatoire (RAM) de largeur prédéterminée correspondant à
40 un bloc d'informations (bi), un ensemble de registres à

décalage ($RDM_1 \dots RDM_j \dots RDM_n$), chacun de capacité correspondant à la largeur de la mémoire, un bus parallèle interne (BUSI) reliant l'accès de la mémoire et les registres à décalage, une logique de sélection d'un registre à décalage (LSR) adaptée pour valider la liaison sur le bus interne entre la mémoire et un registre à décalage prédéterminé, et un ensemble de broches externes d'entrée/sortie (adbloc, admot, numreg, cs, wr, rd, hitbloc, normal/config, hi, di) pour l'entrée d'adresses vers la mémoire (RAM), pour l'entrée d'adresses vers la logique de sélection (LSR), pour l'entrée et la validation de commandes de transfert en lecture ou écriture d'un bloc d'informations (bi) entre la mémoire (RAM) et les registres à décalage (RDM_j), pour l'entrée d'un signal d'horloge vers chaque registre à décalage (RDM_j), pour l'entrée bit à bit d'un bloc d'informations (bi) vers chaque registre à décalage (RDM_j) et pour la sortie bit à bit d'un bloc d'informations de chaque registre à décalage (RDM_j).

23/ - Composant selon la revendication 22, caractérisé en ce qu'il comprend au moins un registre de configuration ($RC_1, RC_2 \dots$) possédant des entrées de programmation, chaque registre de configuration étant relié à une logique de forçage (LF) connectée à la mémoire (RAM) et aux registres à décalage (RDM_j) en vue d'assurer le forçage d'états de ladite mémoire et desdits registres à décalage.

24/ - Composant selon la revendication 23, permettant le choix de la taille des blocs d'informations (bi) traités, caractérisé en ce que :

- la mémoire (RAM) est découpée en zones combinables pour permettre la mémorisation des diverses tailles possibles de blocs d'informations,

- chaque registre à décalage (RDM_j) est découpé en tronçons combinables pour permettre de charger les diverses tailles possibles de blocs d'informations, avec des dérivations aptes à assurer le décalage correspondant à chaque taille,

- le bus interne (BUSI) est doté d'une logique de multiplexage (MT) pour permettre les transferts de blocs d'informations (bi) des diverses tailles entre les

combinaisons de zones de la mémoire (RAM) et les combinaisons correspondantes de tronçons des registres à décalage (RDM_j),

- un registre de configuration (RC_1) est
5 prévu de capacité correspondant au nombre de tailles de blocs possibles,

- la logique de forçage (LF) reliée au
registre (RC_1) comprend une unité logique adaptée pour
commander la logique de multiplexage (MT) en vue de valider
10 les transferts de blocs d'informations (bi) dans une taille
donnée correspondant au paramètre contenu dans le registre de
configuration (RC_1).

25/ - Composant selon l'une des
revendications 23 ou 24, caractérisé en ce que :

15 - l'entrée et la sortie de chaque registre à
décalage (RDM_j) sont reliées à une même broche externe par
l'intermédiaire d'une porte logique (PL_j),

- un registre de configuration (RC_2) est
prévu de capacité correspondant au nombre de registres à
20 décalage (RDM_j),

- la logique de forçage (LF) reliée au
registre de configuration (RC_2) comprend une unité logique
adaptée pour commander les portes logiques (PL_j) en vue de
forcer le fonctionnement de chaque registre à décalage (RDM_j)
25 en mode entrée ou en mode sortie en fonction d'un bit contenu
dans le registre de configuration (RC_2) affecté audit registre
à décalage (RDM_j).

26/ - Composant selon l'une des
revendications 23, 24 ou 25, caractérisé en ce que :

30 - l'entrée et la sortie de chaque registre à
décalage (RDM_j) sont reliées à une même broche externe par
l'intermédiaire d'une porte logique (PL_j),

- un registre de configuration (RC_3) est
prévu de capacité correspondant au nombre de registres à
35 décalage (RDM_j),

- la logique de forçage (LF) reliée au
registre de configuration (RC_3) comprend une unité logique
reliée à la commande de lecture de la mémoire (RAM) et adaptée
pour commander chaque porte logique (PL_j) soit en mode sortie
40 au moment de la lecture de la mémoire (transfert de la

mémoire RAM vers le registre correspondant RDM_j) pendant toute la durée du vidage dudit registre à décalage (RDM_j), soit en mode entrée, le reste du temps.

5 27/ - Composant selon l'une des revendications 23, 24, 25 ou 26, caractérisé en ce qu'il comprend une broche externe d'entrée (bit/bloc), une ou des broches externes d'entrée/sortie d'informations (data), une logique de commande (COM) reliée à la broche d'entrée
10 (bit/bloc), aux broches d'entrée/sortie (data), à la mémoire (RAM) et à la logique de sélection (LSR) et adaptée selon l'état de l'entrée (bit/bloc) pour engendrer soit les transferts de blocs d'informations (bi) entre mémoire (RAM) et registres à décalage (RDM_j), soit des transferts de bits
15 directement entre la mémoire (RAM) et les broches (data).

28/ - Composant selon les revendications 23 et 27 prises ensemble, caractérisé en ce que les registres de configuration (RC_1 , $RC_2...$) sont reliés :

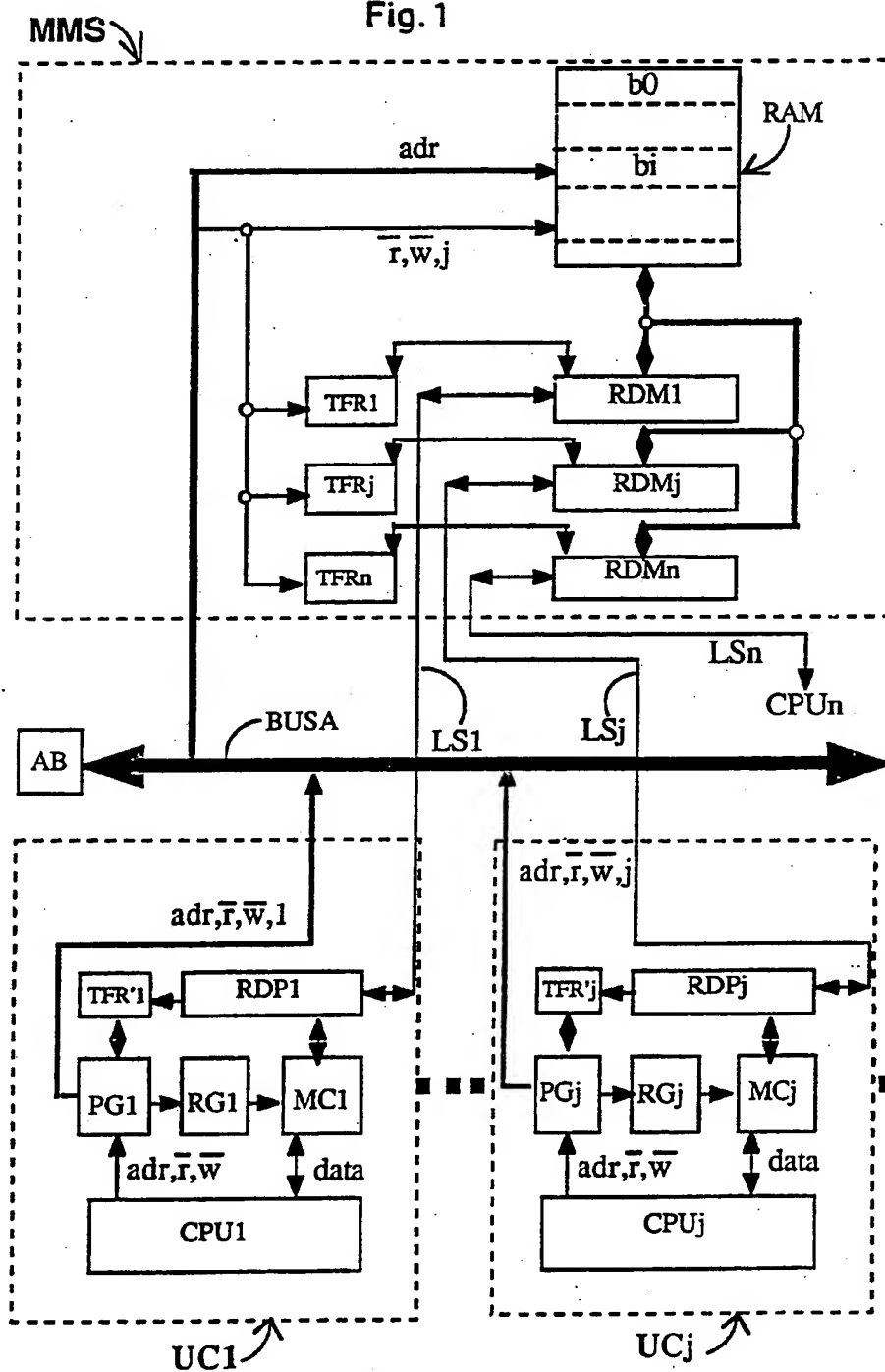
- d'une part à la logique de sélection (LSR)
20 laquelle est adaptée pour sélectionner lesdits registres de configuration pour des adresses prédéterminées affectées auxdits registres,

- d'autre part, à la logique de commande (COM) laquelle est adaptée pour transmettre les
25 données en provenance des broches d'entrée/sortie (data) vers lesdits registres de configuration en vue de leur programmation.

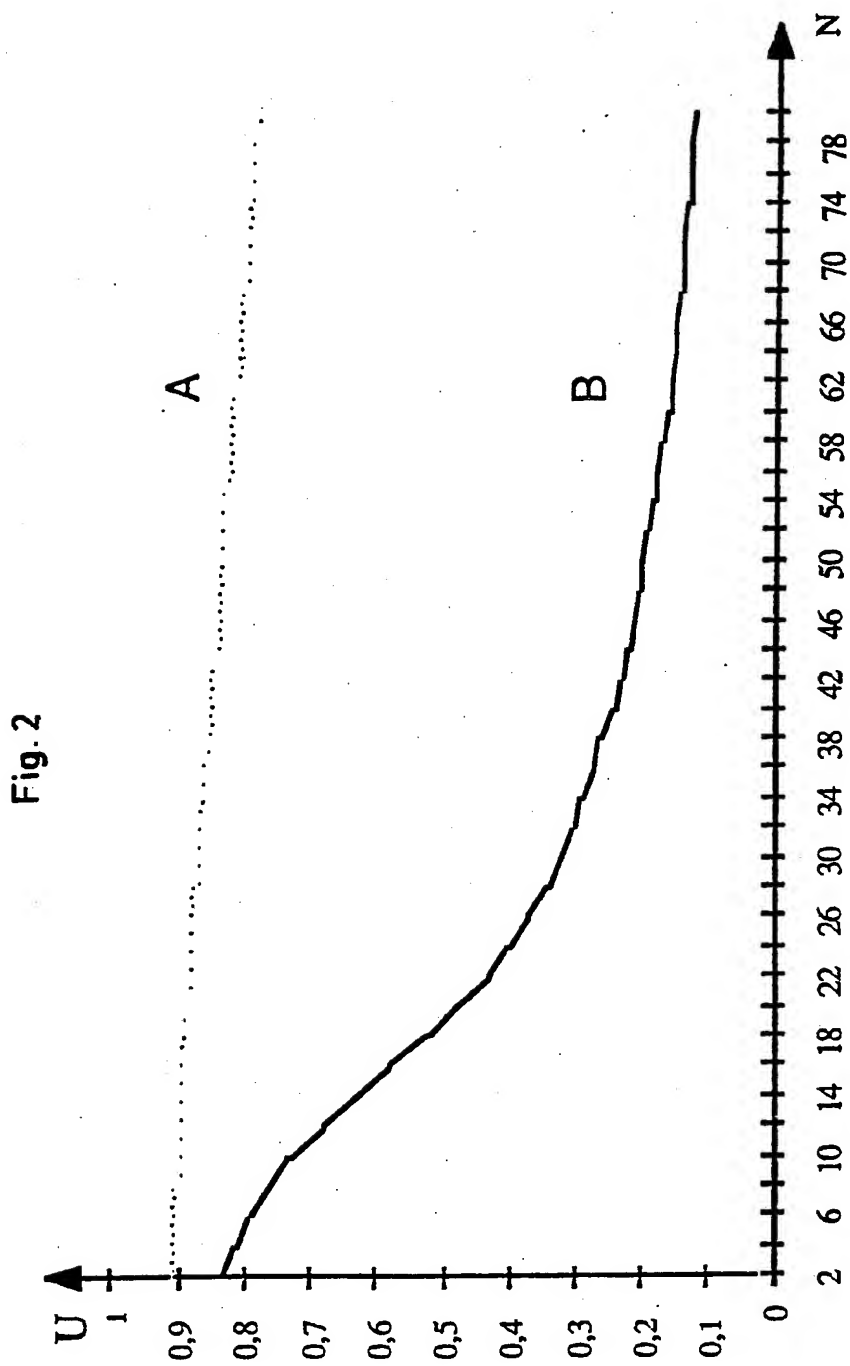
29/ - Composant selon l'une des revendications 22 à 28, dans lequel sur le bus interne (BUSI)
30 reliant l'accès de la mémoire (RAM) et les registres à décalage (RDM_j) est interposée une logique de type "barillet" (BS) ("barrel shifter") apte à assurer une permutation circulaire sur les bits de chaque bloc d'informations, ladite
logique (BS) possédant une entrée de commande du pas de
35 glissement en unité mot connectée à des broches d'entrée (admot).

1/22

Fig. 1

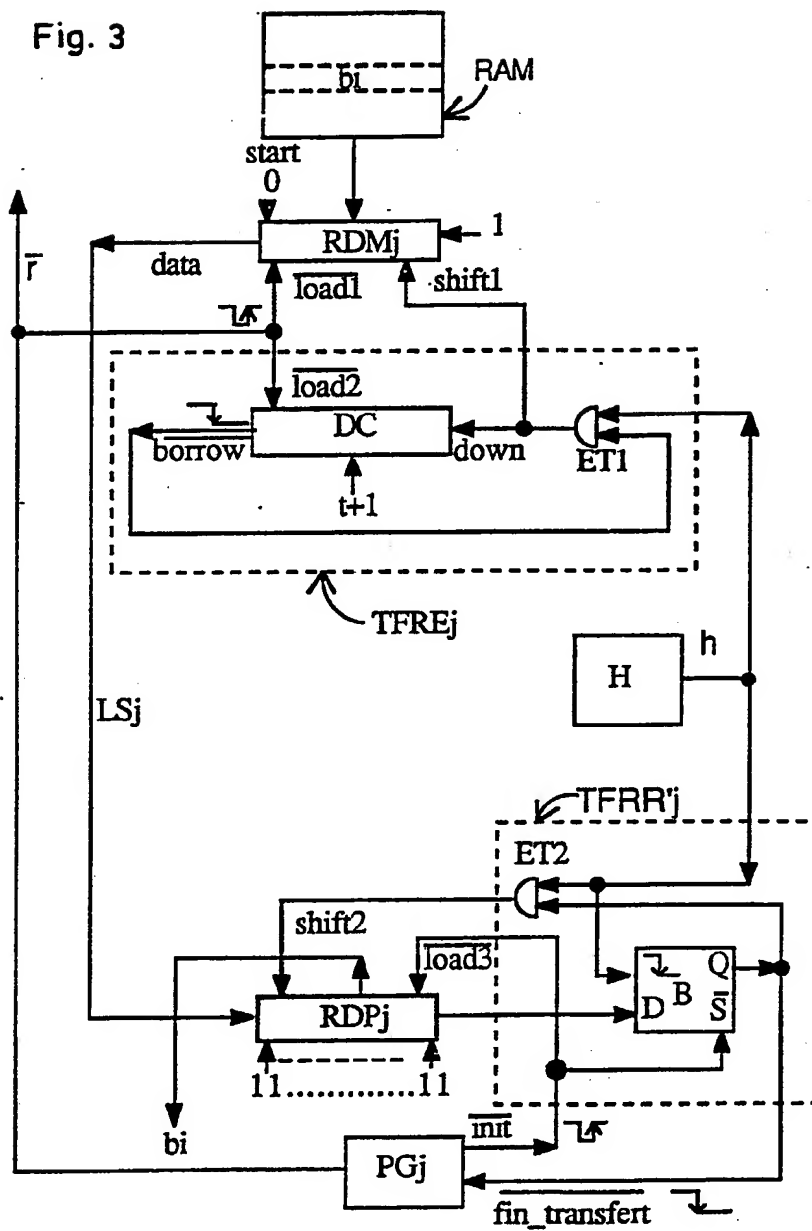


2/22



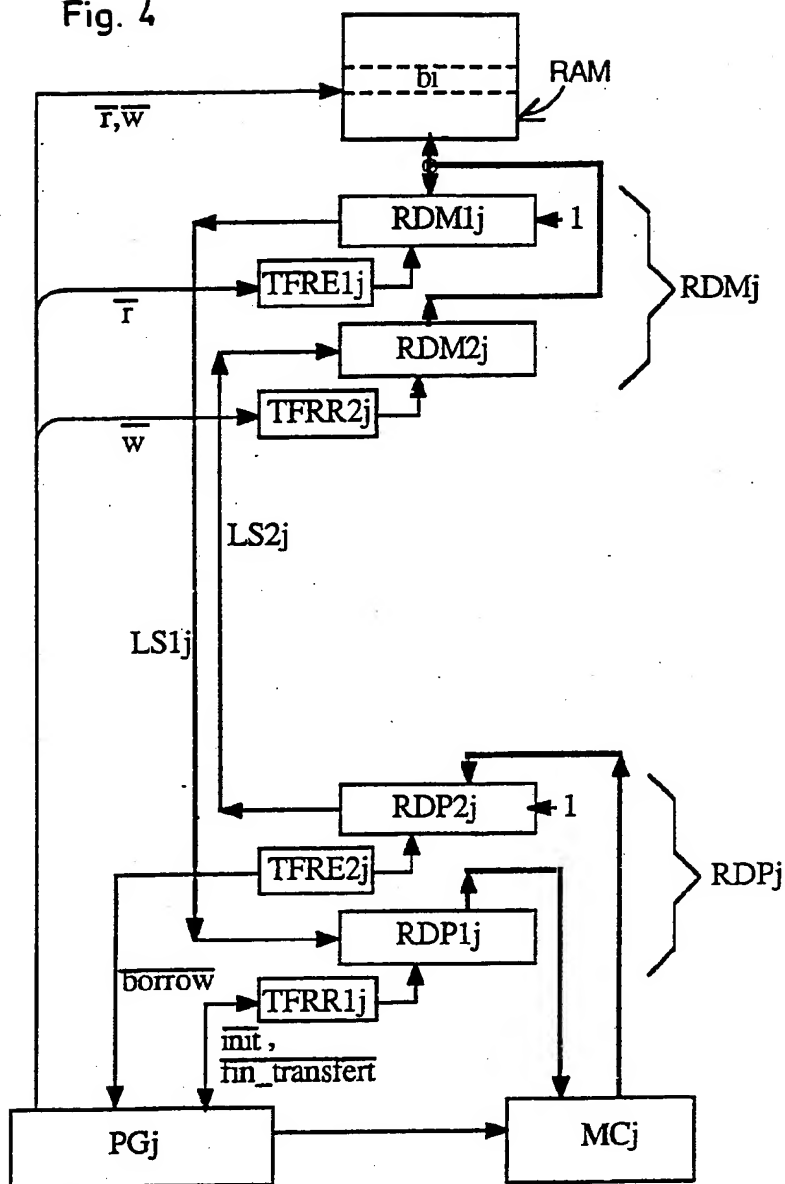
3/22

Fig. 3



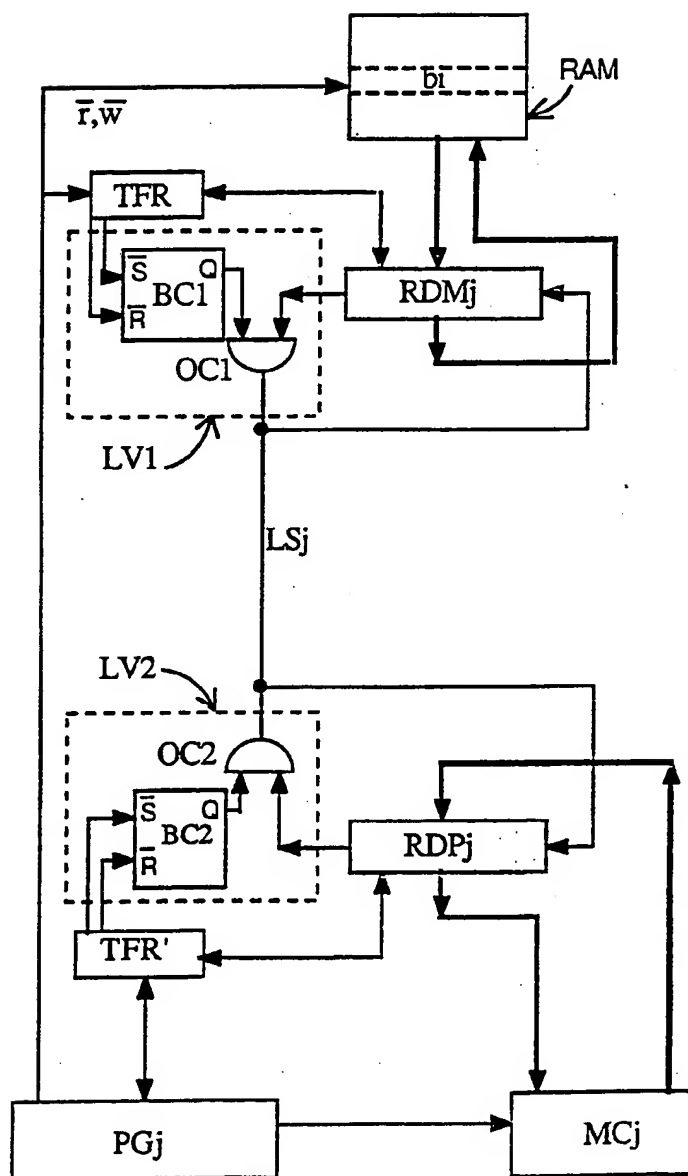
4 / 22

Fig. 4



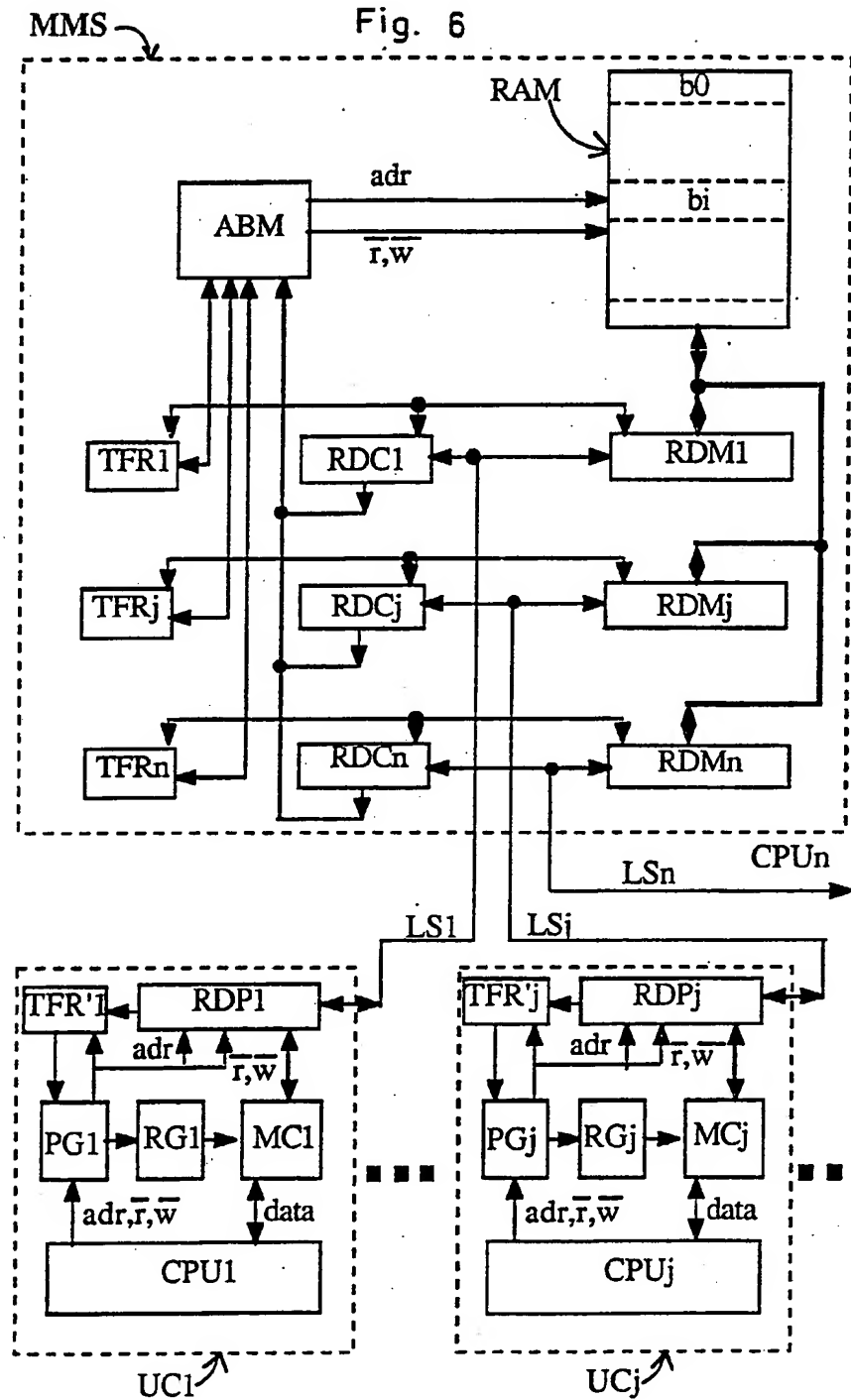
5/22

Fig. 5



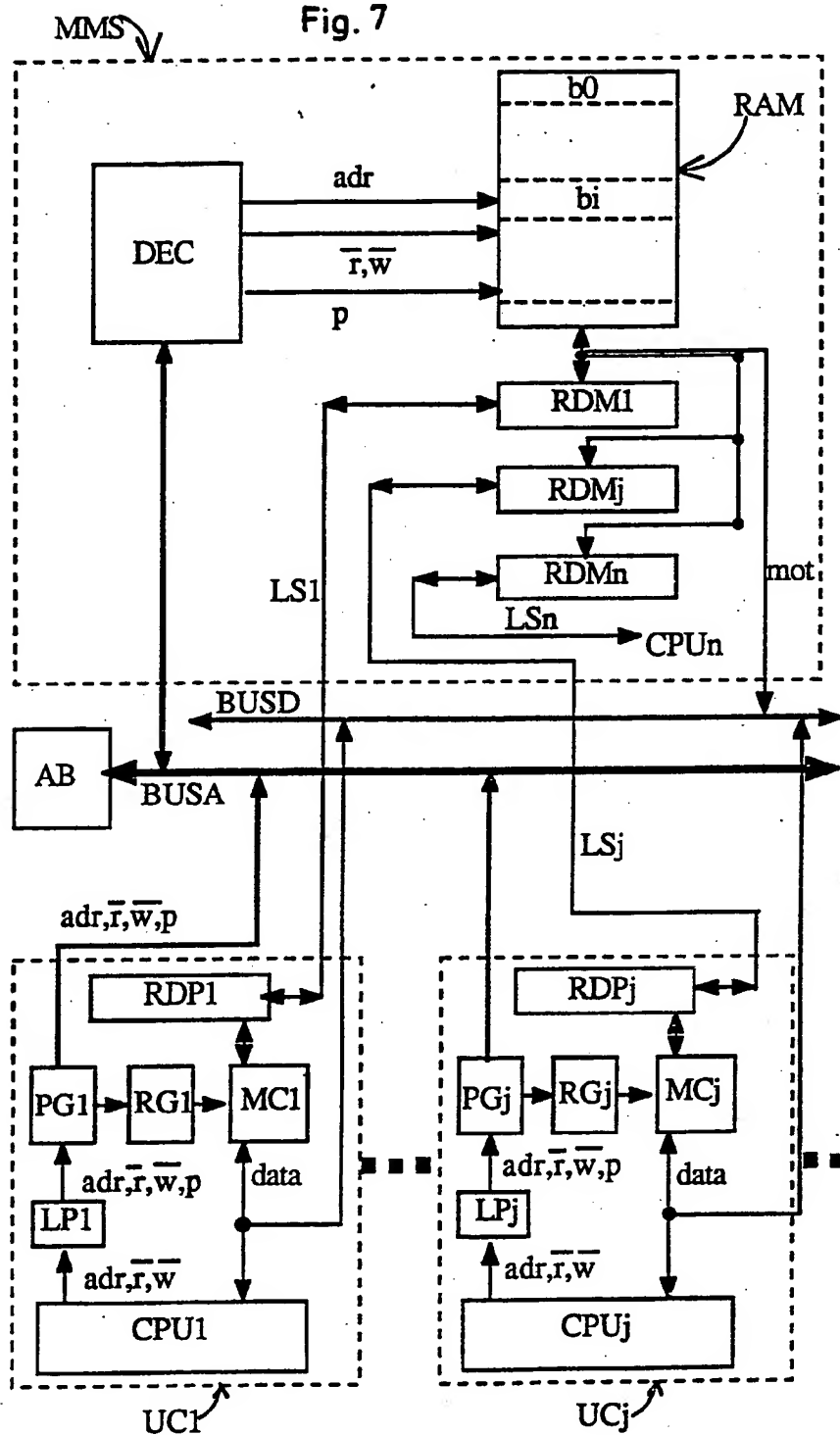
6 / 22

Fig. 6



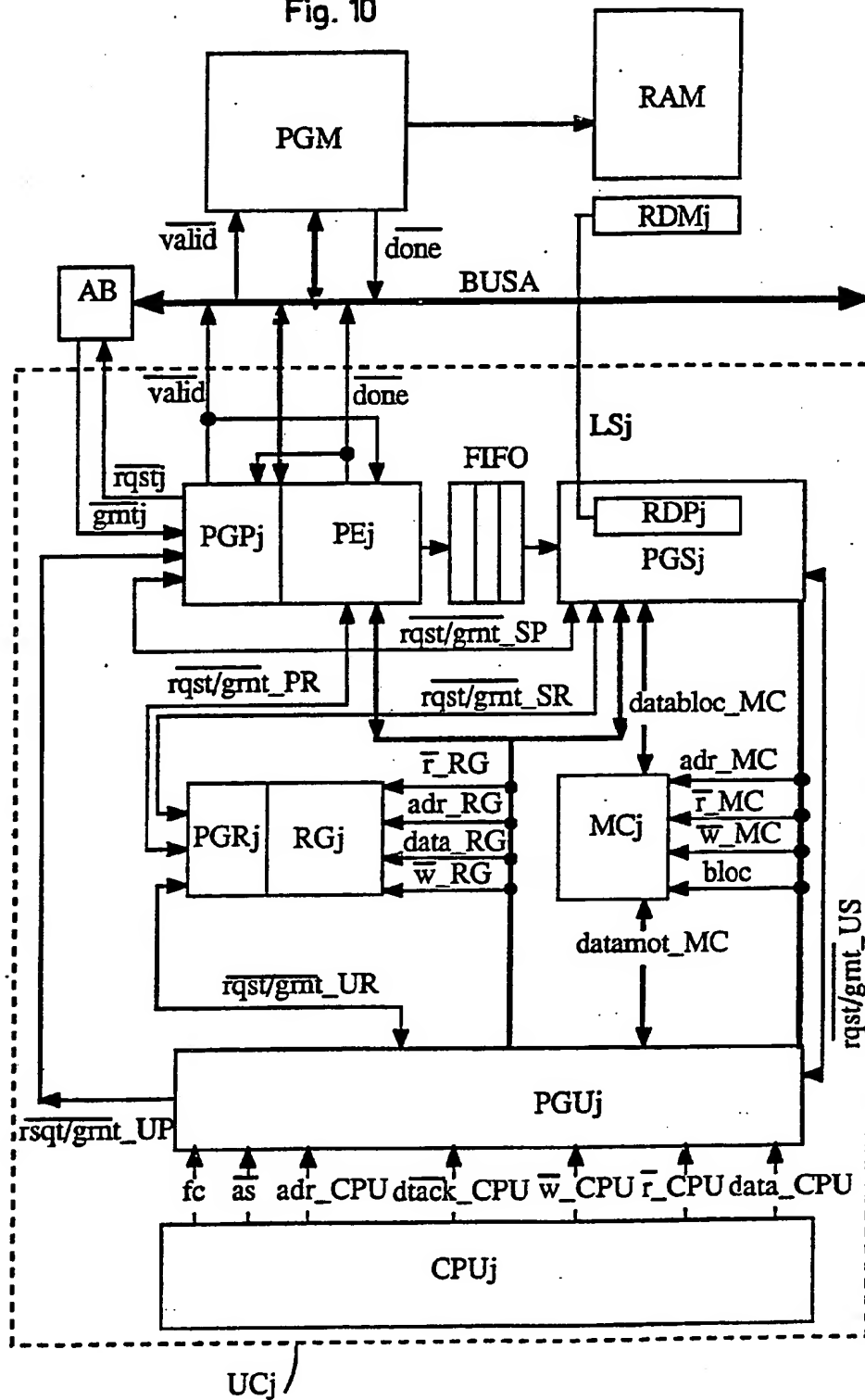
7/22

Fig. 7



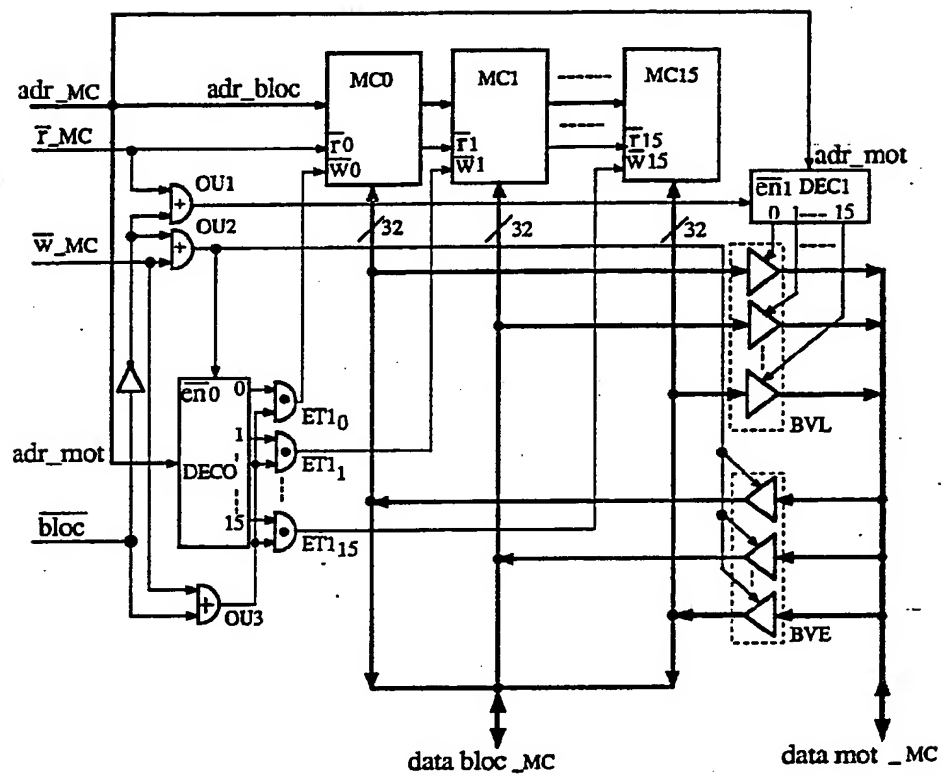
10/22

Fig. 10



11/22

Fig. 11



12/22

Fig. 12a

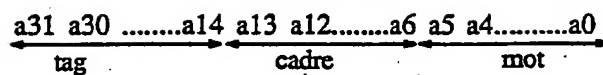


Fig. 12b

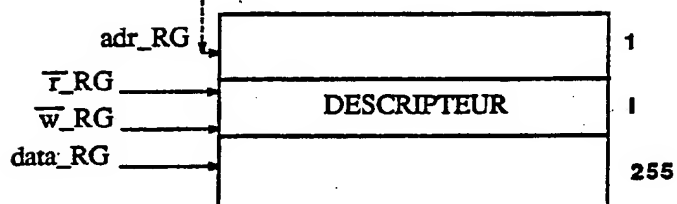
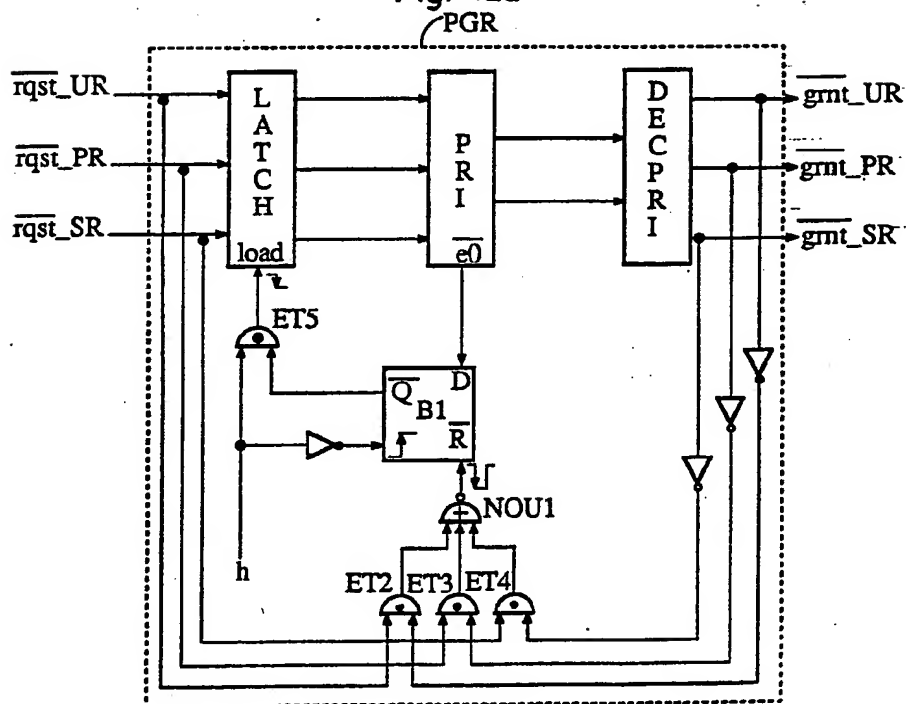


Fig. 12c

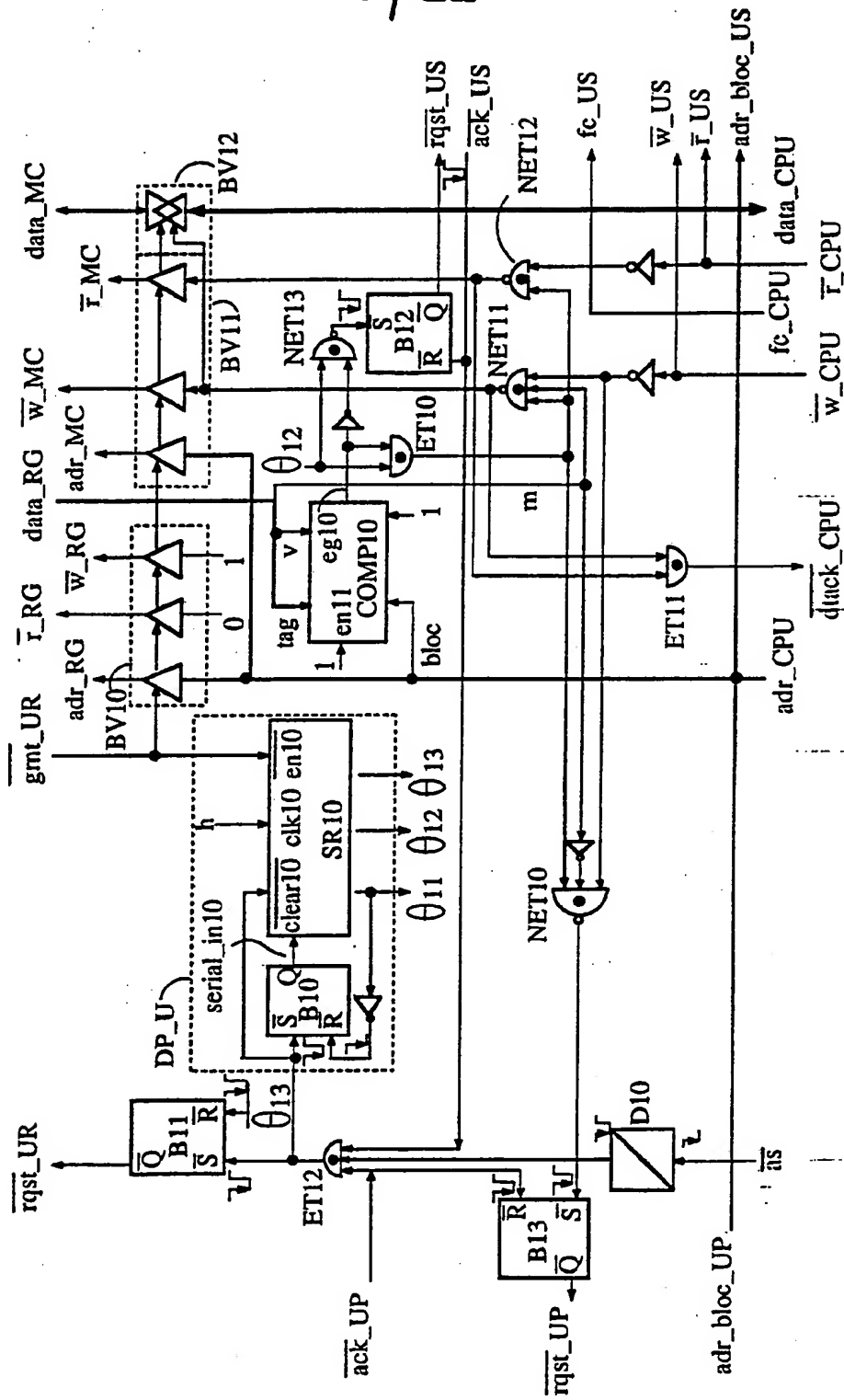


Fig. 12d



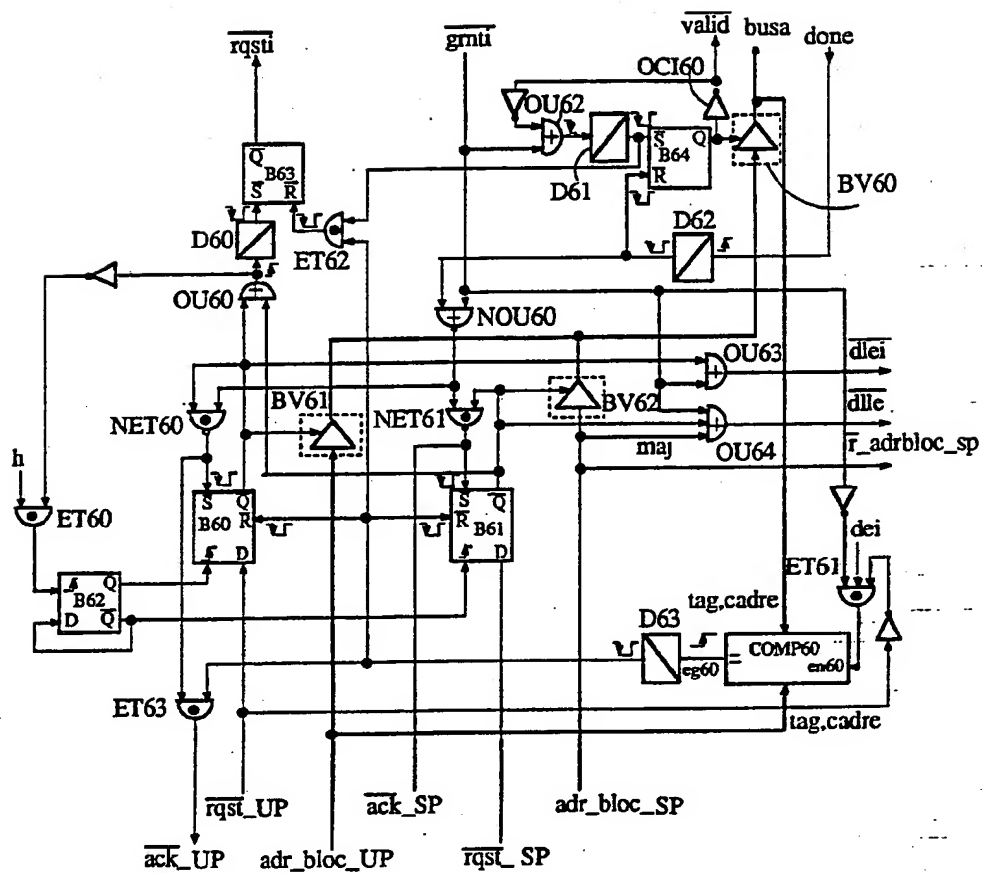
13/22

Fig. 13



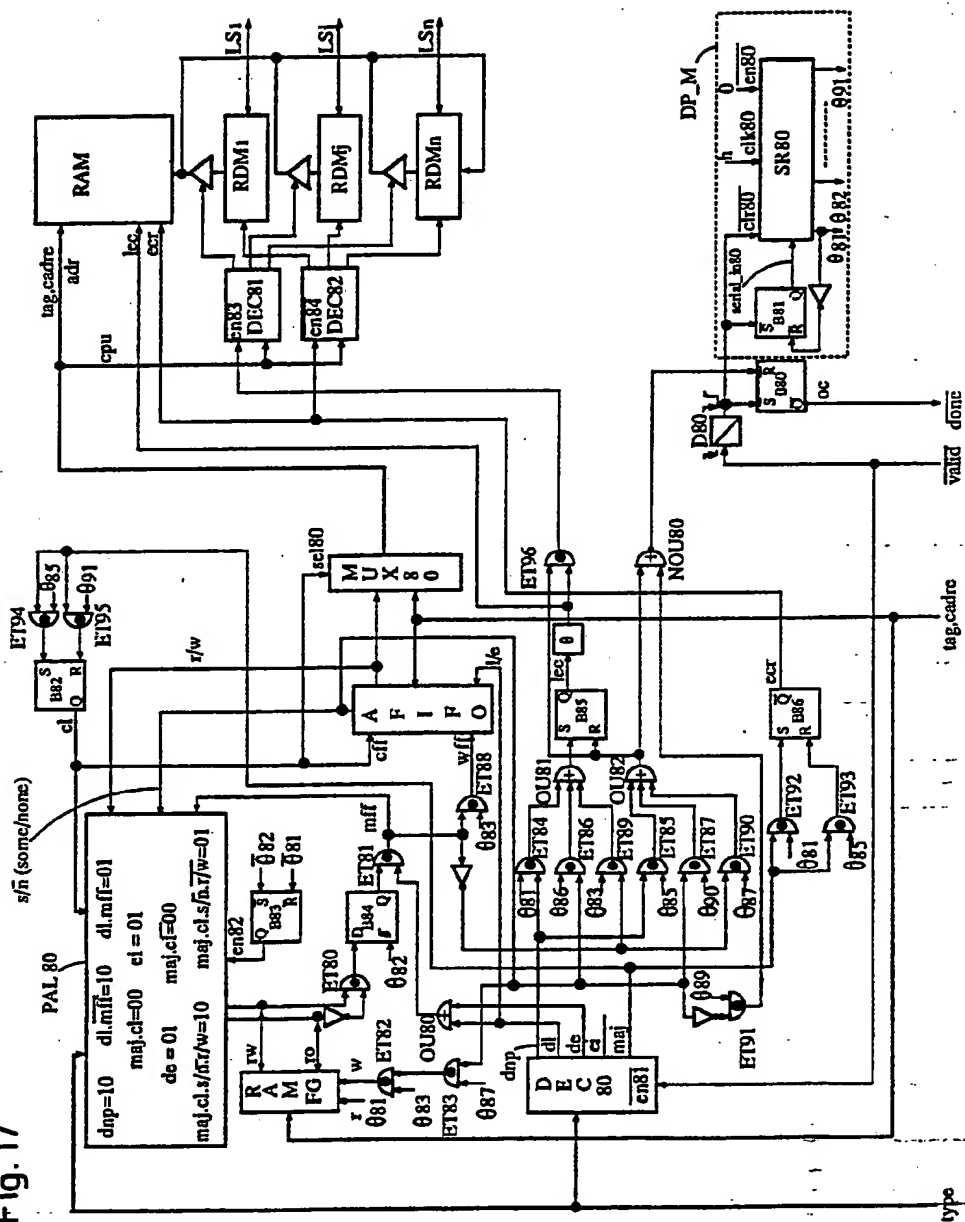
16/22

Fig. 16

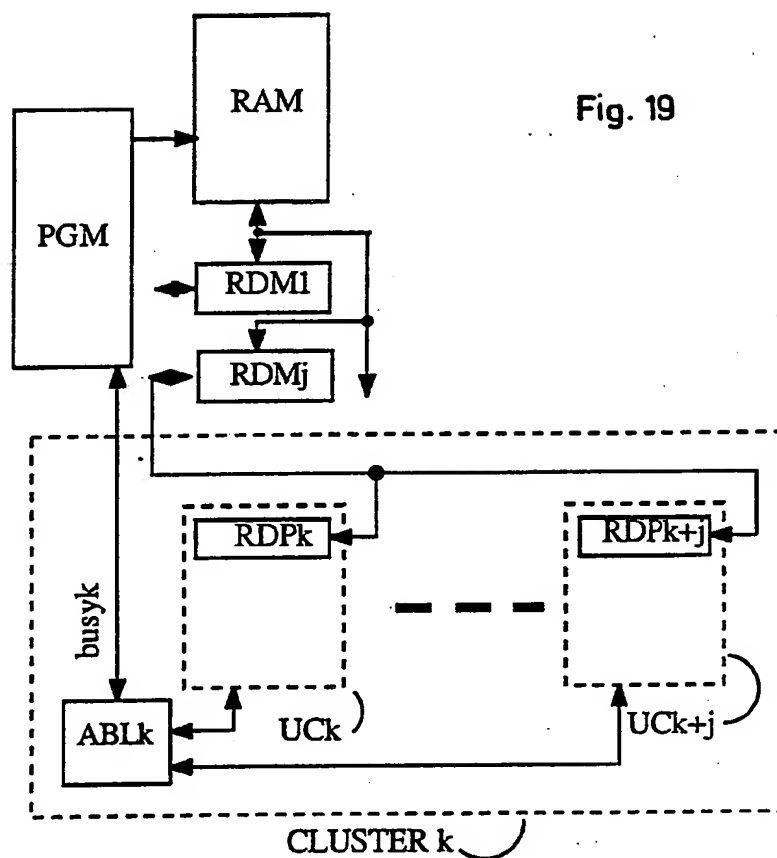


17 / 22

Fig. 17



19 / 22



20/22

FIG. 20 a

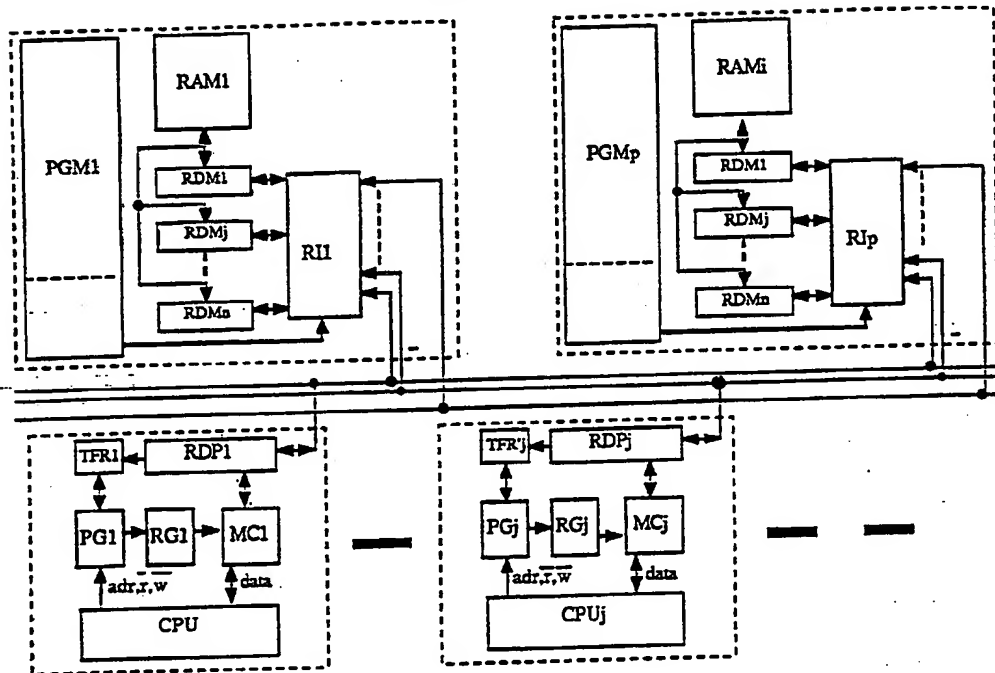
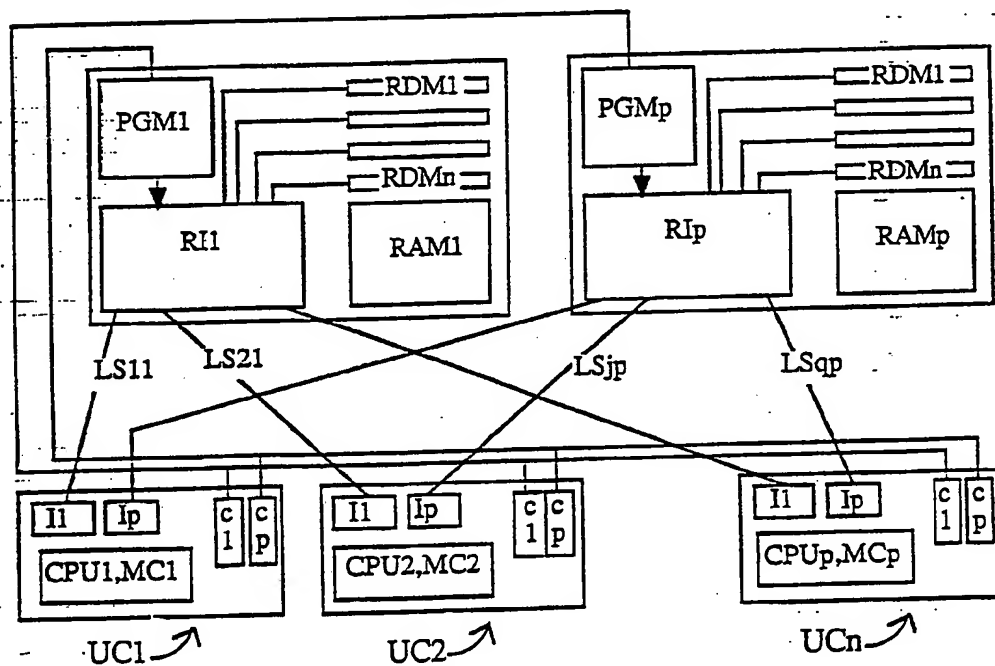
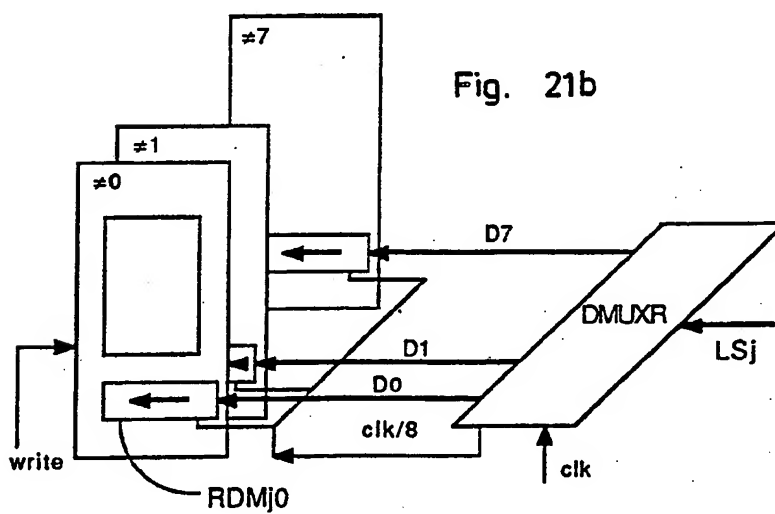
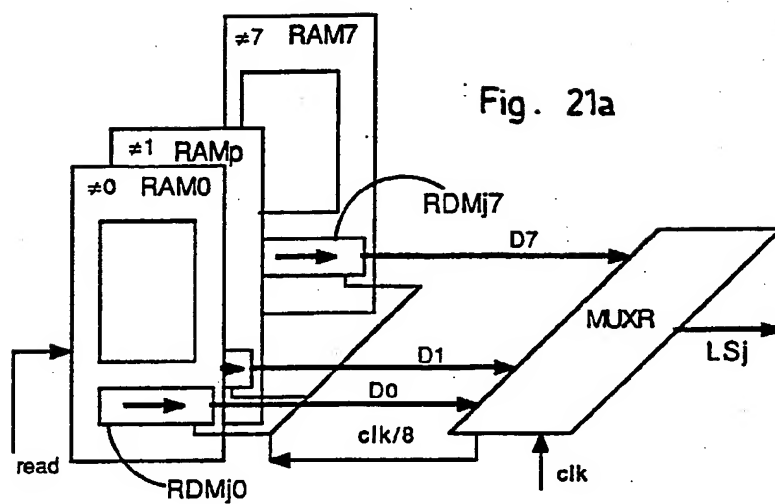


FIG. 20 b

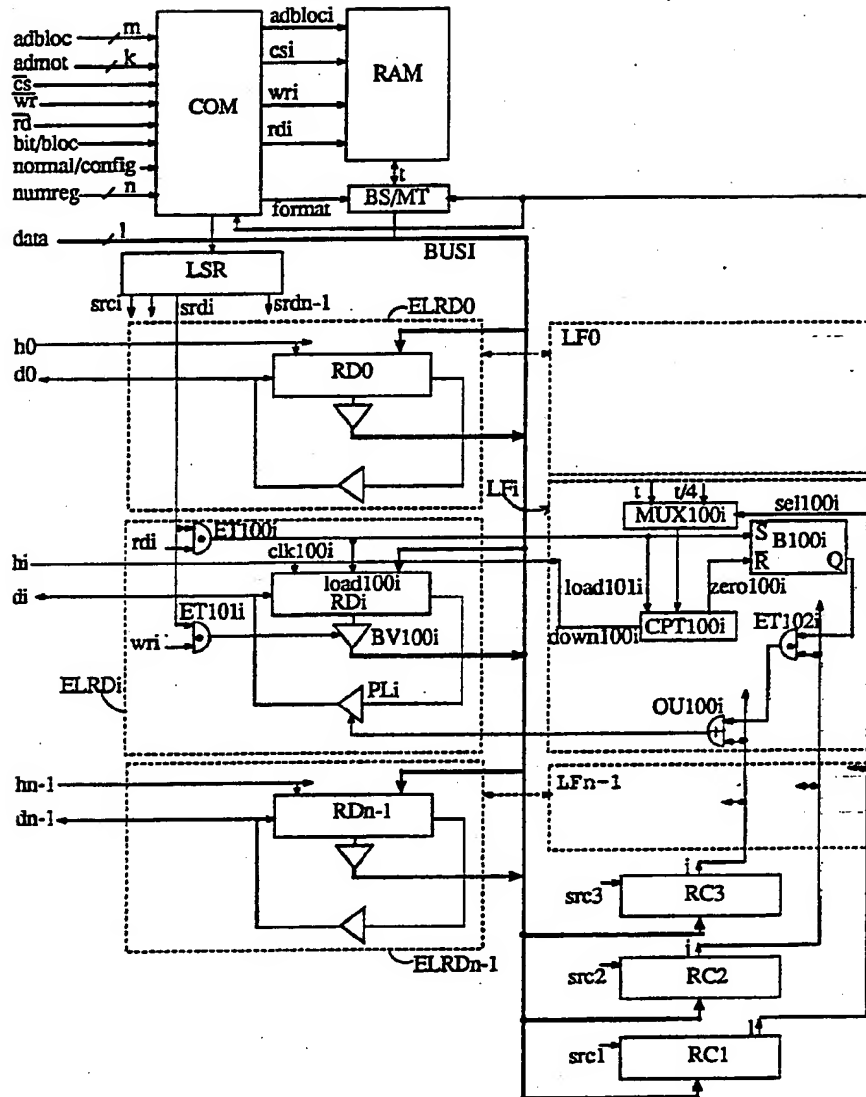


21/22



22/22

Fig.22



INTERNATIONAL SEARCH REPORT

International Application No PCT/FR 88/00608

I. CLASSIFICATION OF SUBJECT MATTER (If several classification symbols apply, indicate all) *		
According to International Patent Classification (IPC) or to both National Classification and IPC		
Int. Cl. ⁴ G 06 F 12/08; G 11 C 8/00		
II. FIELDS SEARCHED		
Minimum Documentation Searched *		
Classification System	Classification Symbols	
Int. Cl. ⁴	G 06 F 12/08; G 11 C 8/00	
Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched *		
III. DOCUMENTS CONSIDERED TO BE RELEVANT*		
Category *	Citation of Document, ¹¹ with Indication, where appropriate, of the relevant passages ¹²	Relevant to Claim No. ¹³
A	EP, A, 0187289 (IBM) 16 July 1986 see column 2, lines 8-20, fig. 3	1,20-22
A	WO, A, 82/02615 (WESTERN ELECTRIC) 5 August 1982 see page 3, line 31 - page 4, line 22; fig. 2	1,20-22
A	EP, A, 0166192 (IBM) 2 January 1986 see page 3, line 21 - page 4, line 10; page 6, lines 19-28; fig. 2	1,20-22
A	IEEE Transactions on Computers, vol. C-31, No. 11, November 1982, IEEE, (New York, US), M Dubois et al.: "Effescts of cache coherency in multiprocessors", pages 1083-1099 see paragraphs I,II,III	6-11
A	EP, A, 0126976 (IBM) 5 December 1984 see page 8, line 15 - page 9, line 13; fig. 5	1,20-22
<p>* Special categories of cited documents: ¹⁰</p> <p>"A" document defining the general state of the art which is not considered to be of particular relevance</p> <p>"E" earlier document but published on or after the international filing date</p> <p>"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)</p> <p>"O" document referring to an oral disclosure, use, exhibition or other means</p> <p>"P" document published prior to the international filing date but later than the priority date claimed</p> <p>"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention</p> <p>"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step</p> <p>"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.</p> <p>"&" document member of the same patent family</p>		
IV. CERTIFICATION		
Date of the Actual Completion of the International Search	Date of Mailing of this International Search Report	
8 March 1989 (08.03.89)	29 March 1989 (29.03.89)	
International Searching Authority	Signature of Authorized Officer	
EUROPEAN PATENT OFFICE		

**ANNEX TO THE INTERNATIONAL SEARCH REPORT
ON INTERNATIONAL PATENT APPLICATION NO.**

FR 8800608
SA 25887

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report. The members are as contained in the European Patent Office EDP file on 21/03/89. The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP-A- 0187289	16-07-86	JP-A- 61161562 US-A- 4633440	22-07-86 30-12-86
WO-A- 8202615	05-08-82	EP-A, B 0069764 US-A- 4412313 CA-A- 1168376 DE-A- 3176797	19-01-83 25-10-83 29-05-84 28-07-88
EP-A- 0166192	02-01-86	JP-A- 61020157	28-01-86
EP-A- 0126976	05-12-84	JP-A- 59216269 US-A- 4616310	06-12-84 07-10-86


EP-A FORM 10/79

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

SS0022002304

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale N° PCT/FR 88/00608

I. CLASSEMENT DE L'INVENTION (si plusieurs symboles de classification sont applicables, les indiquer tous) ⁷ Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB CIB ⁴ : G 06 F 12/08; G 11 C 8/00		
II. DOMAINES SUR LESQUELS LA RECHERCHE A PORTÉ Documentation minimale consultée ⁸		
Système de classification	Symboles de classification	
CIB ⁴	G 06 F 12/08; G 11 C 8/00	
Documentation consultée autre que la documentation minimale dans la mesure où de tels documents font partie des domaines sur lesquels la recherche a porté ⁹		
III. DOCUMENTS CONSIDÉRÉS COMME PERTINENTS ¹⁰		
Catégorie [*]	Identification des documents cités, ¹¹ avec indication, si nécessaire, des passages pertinents ¹²	N° des revendications visées ¹³
A	EP, A, 0187289 (IBM) 16 juillet 1986 voir colonne 2, lignes 8-20, figure 3 --	1,20-22
A	WO, A, 82/02615 (WESTERN ELECTRIC) 5 août 1982 voir page 3, ligne 31 - page 4, ligne 22; figure 2 --	1,20-22
A	EP, A, 0166192 (IBM) 2 janvier 1986 voir page 3, ligne 21 - page 4, ligne 10; page 6, lignes 19-28; figure 2 --	1,20-22
A	IEEE Transactions on Computers, volume C-31, no. 11, novembre 1982, IEEE, (New York, US), M. Dubois et al.: "Effects of cache coherency in multiprocessors", pages 1083-1099 voir paragraphes I,II,III --	6-11
<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>[*] Catégories spéciales de documents cités: ¹¹</p> <p>« A » document définissant l'état général de la technique, non considéré comme particulièrement pertinent</p> <p>« E » document antérieur, mais publié à la date de dépôt international ou après cette date</p> <p>« L » document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée)</p> <p>« O » document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens</p> <p>« P » document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée</p> </div> <div style="width: 50%;"> <p>« T » document ultérieur publié postérieurement à la date de dépôt international ou à la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention</p> <p>« X » document particulièrement pertinent: l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive</p> <p>« Y » document particulièrement pertinent: l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier.</p> <p>« & » document qui fait partie de la même famille de brevets</p> </div> </div>		
IV. CERTIFICATION		
Date à laquelle la recherche internationale a été effectivement achevée 8 mars 1989	Date d'expédition du présent rapport de recherche internationale <div style="text-align: right;">20 MAR 1989</div>	
Administration chargée de la recherche internationale OFFICE EUROPEEN DES BREVETS	Signature du fonctionnaire autorisé <div style="text-align: right;">  P. G. VAN DER PUTTEN </div>	

III. DOCUMENTS CONSIDÉRÉS COMME PERTINENTS		
(SUITE DES RENSEIGNEMENTS INDICUÉS SUR LA DEUXIÈME FEUILLE)		
Catégorie *	Identification des documents cités, avec indication, si nécessaire, des passages pertinents	N° des revendications visées
A	EP, A, 0126976 (IBM) 5 décembre 1984 voir page 8, ligne 15 - page 9, ligne 13; figure 5 -----	1,20-22

**ANNEXE AU RAPPORT DE RECHERCHE INTERNATIONALE
RELATIF A LA DEMANDE INTERNATIONALE NO.**

FR 8800608
SA 25887

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche internationale visé ci-dessus.
Lesdits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du 21/03/89
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets.

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
EP-A- 0187289	16-07-86	JP-A- 61161562 US-A- 4633440	22-07-86 30-12-86
WO-A- 8202615	05-08-82	EP-A, B 0069764 US-A- 4412313 CA-A- 1168376 DE-A- 3176797	19-01-83 25-10-83 29-05-84 28-07-88
EP-A- 0166192	02-01-86	JP-A- 61020157	28-01-86
EP-A- 0126976	05-12-84	JP-A- 59216269 US-A- 4616310	06-12-84 07-10-86

EPO FORM P0812

Pour tout renseignement concernant cette annexe : voir Journal Officiel de l'Office européen des brevets, No.12/82

SS0022002307

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

☐ BLACK BORDERS

☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES

☒ FADED TEXT OR DRAWING

☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING

☐ SKEWED/SLANTED IMAGES

☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS

☐ GRAY SCALE DOCUMENTS

☒ LINES OR MARKS ON ORIGINAL DOCUMENT

☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY

☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.